Secure Starter Kit Cloud Connect Installation & Setup Guide

Date: December 08, 2020 | Version 1.0
FINAL

CONTENTS

1 INTRODUCTION

1.1 Purpose of the Document

The Cloud Connect Installation & Setup Guide provides an overview of the AWS services required to run the demo's provided in the Security Starter Quick Start Guides, as well as detailed instructions to setup and configure those required services. Each of these services MUST be setup and configured (only once), prior to running the demo's outlined in the Security Starter Quick Start Guides.

1.2 Prerequisites, Background information & AWS Cloud Services Descriptions

1. AWS Account Management Console – the user will need to create their own AWS Account and is used as the basis for the configuration of the other services required to run the demo's provided in the Security Starter Kits. The creation of an account provides the following access and feature;
    - Discover and experiment with over 150 AWS services, many of which you can try for free.
    - Build your cloud-based applications in any AWS data center throughout the world.
    - Manage and monitor users, service usage, health, and monthly billing.
    - Get in-console help from AWS Support.
    - *Link to create AWS Account;* https://portal.aws.amazon.com/billing/signup#/start

2. AWS EC2 Instance Service – https://aws.amazon.com/ec2

    Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers. Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment.

    - *The output of the setup and configuration of the EC2 instance will provide the user with URL and Login credentials required to run their own instance of the Security Starter Kit Cloud Connect Tool.*

3. AWS Relational Database Service (Amazon RDS) – https://aws.amazon.com/rds/

    Makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups. It frees you to focus on your applications so you can give them the fast performance, high availability, security and compatibility they need.

    Amazon RDS is available on several database instance types - optimized for memory, performance or I/O - and provides you with six familiar database engines to choose from, including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database, and SQL Server. You can use the AWS Database Migration Service to easily migrate or replicate your existing databases to Amazon RDS.

    - *MySQL is employed as the database instance type when configuring Amazon RDS service.*

4. Docker Hub - Cloud-based application registry and development team collaboration services.
   https://www.docker.com/
   https://hub.docker.com/

   Docker Hub is the world's largest repository of container images with an array of content sources including container community developers, open source projects and independent software vendors (ISV) building and distributing their code in containers. Users get access to free public repositories for storing and sharing images or can choose subscription plan for private repos.

   - *Docker is the repository service used to store source code for our web-based, open-source "Security Starter Kit Cloud Connect Tool". The user will need to update, configure and build an "Image" from the source code stored on Docker Hub using their specific AWS Account and AWS Service credentials. These instructions are provided below in the document.*

5. AWS OTA Role Access - https://docs.aws.amazon.com/freertos/latest/userguide/create-ota-user-policy.html

   When you create an OTA update, the OTA Update Manager service creates an AWS IoT job to notify your devices that an update is available. The OTA demo application runs on your device and creates a FreeRTOS task that subscribes to notification topics for AWS IoT jobs and listens for update messages. When an update is available, the OTA Agent publishes requests to AWS IoT and receives updates using the HTTP or MQTT protocol, depending on the settings you chose. The OTA Agent checks the digital signature of the downloaded files and, if the files are valid, installs the firmware update. If you don't use the FreeRTOS OTA Update demo application, you must integrate the OTA Agent library into your own application to get the firmware update capability.

   - *OTA setup and configuration is listed in Section 5 of the SSK Cloud Connect Users Guide.*
   - *These steps do not need to be completed to run the demo outlined in the Quick Start Guide, but will need to be configured in order to perform OTA firmware updates from within AWS Cloud Services.*

2 AWS ACCOUNT CREATION & SETUP EC2 SERVICE

2.1 Login or Create your AWS Account

Note: If the User does not have an AWS Account, you will need to create one and this is used as the basis for the configuration of the other services required to run the demo's provided in the Security Starter Kits.

Login URL: https://aws.amazon.com/console/

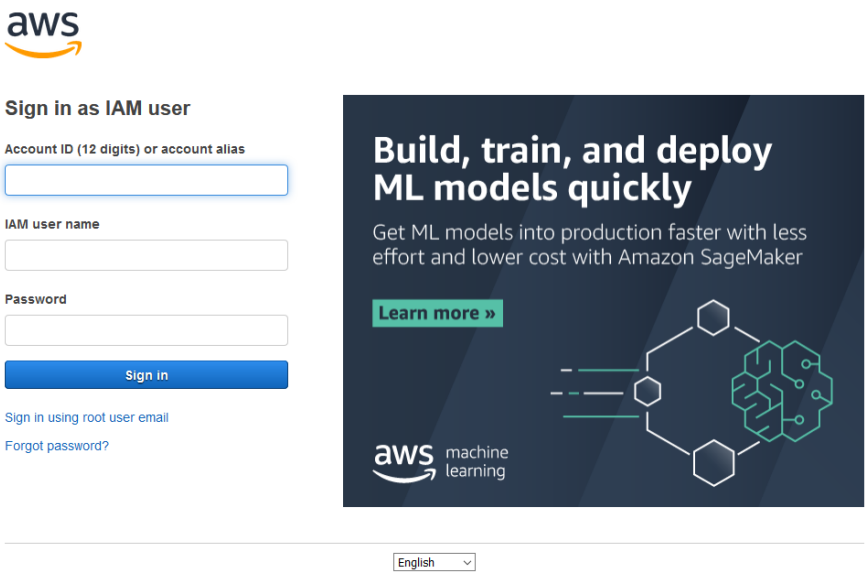


Figure 1: Login page

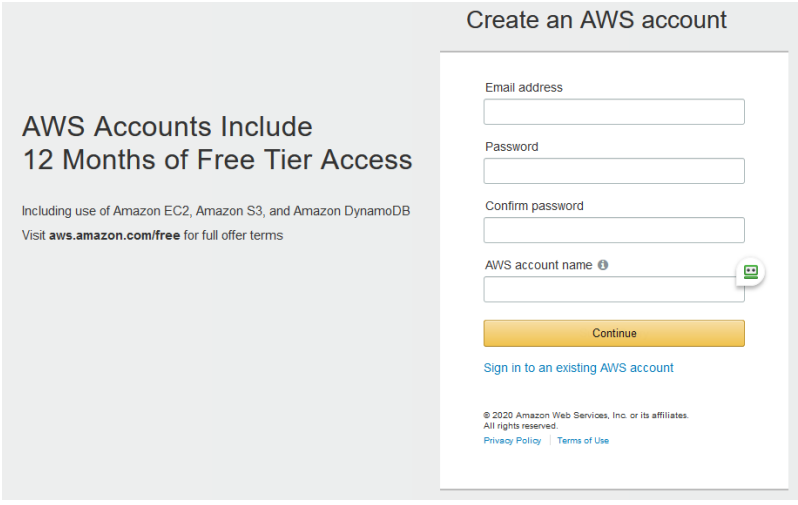


Figure 2: Create New Account page

2.2 AWS EC2 Instance Service

- Go to AWS Console >> Services >> Select EC2 (Under Compute section).

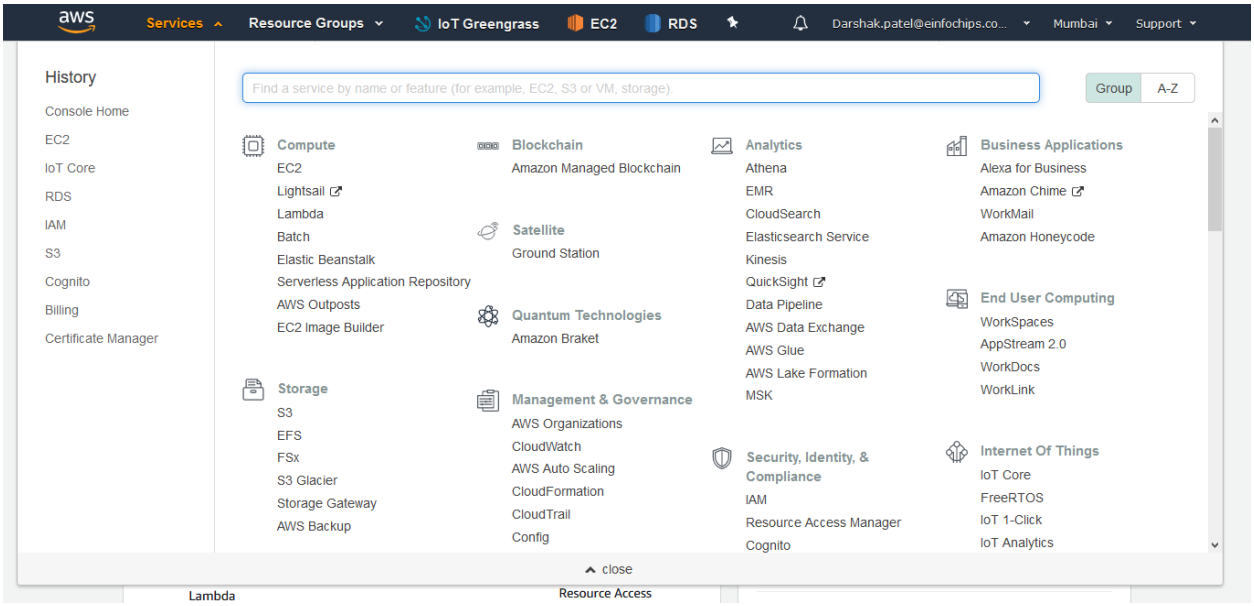


Figure 2: Select EC2 Instance

2.3 EC2 Dashboard

- Go to Instances >> Instances Click on it.

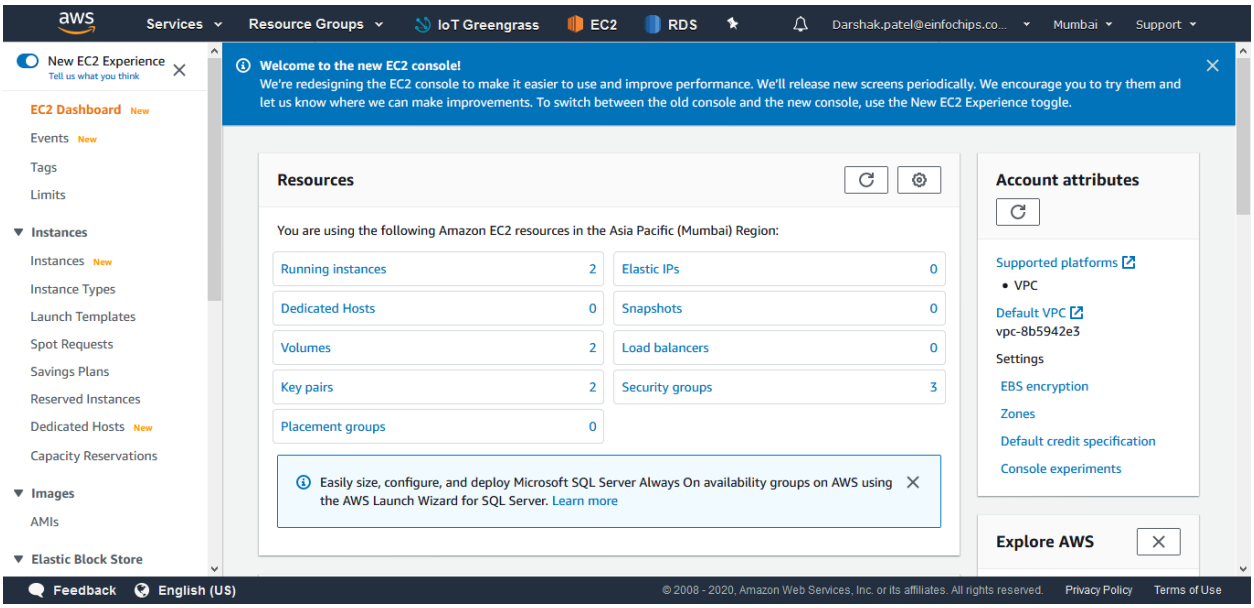


Figure 2: List EC2 Dashboard

2.4 Creating an EC2 Instance

Step 1: Click on Launch instances (Top right corner)

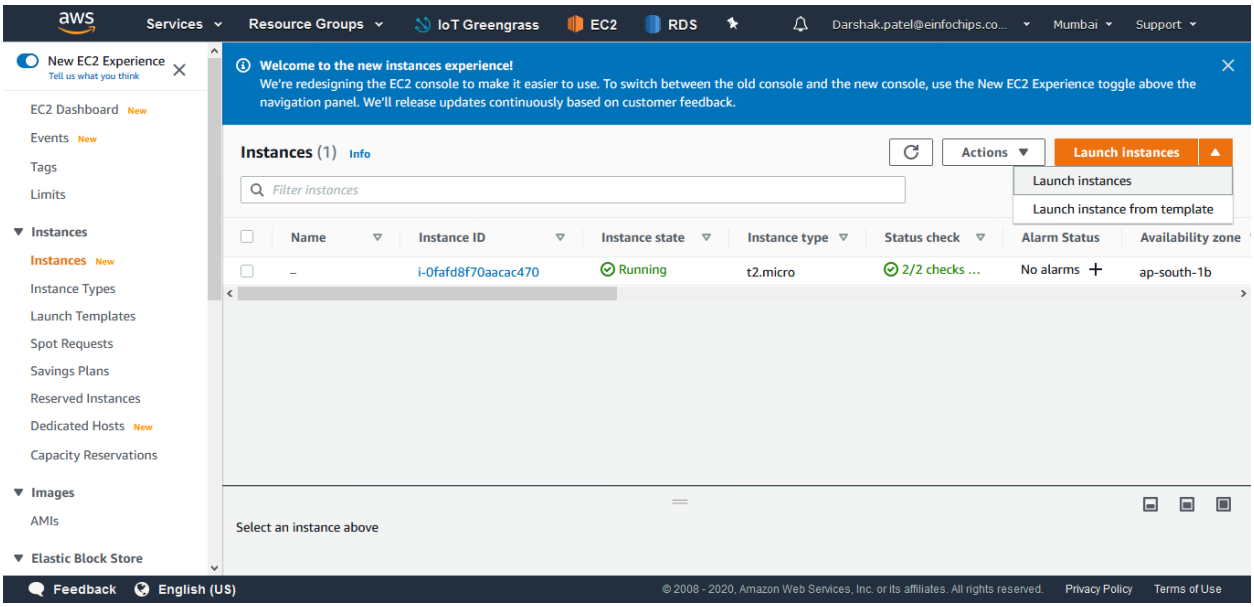


Figure 3: Launch EC2 Instance

- Choose an Amazon Machine Image
  Search “Ubuntu Server 18.04 LTS” in textbox then press select button.

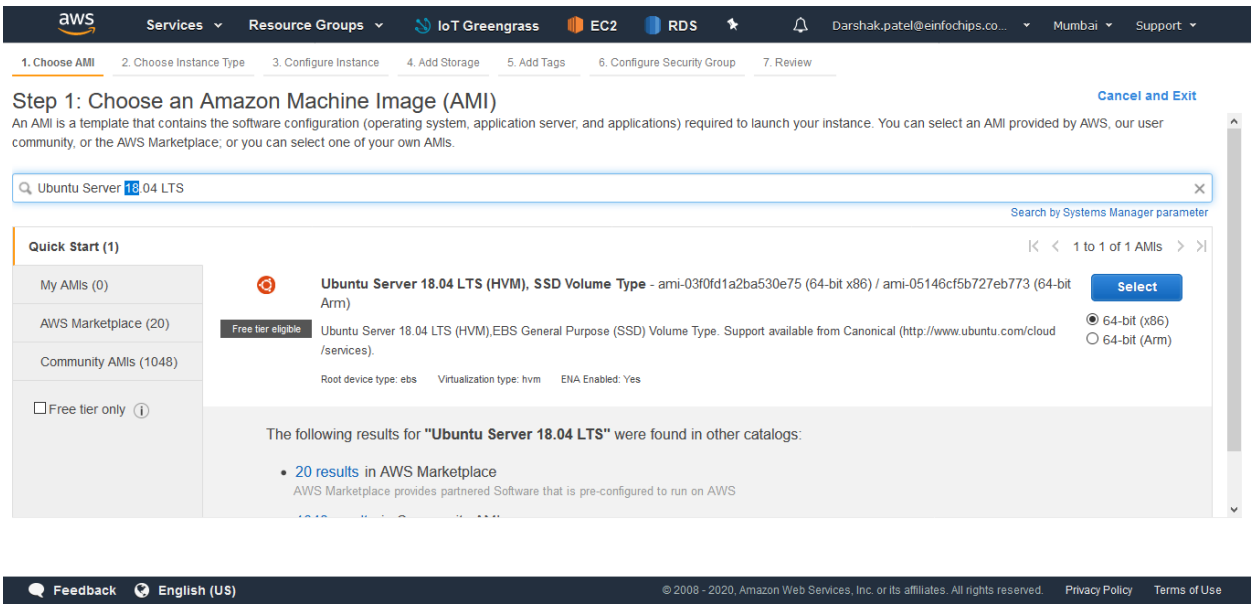


Figure 4: Ubuntu AMI

Step 2: Choose an Instance Type (Change as per your performance requirement)

- Click on Next: Configure Instance details

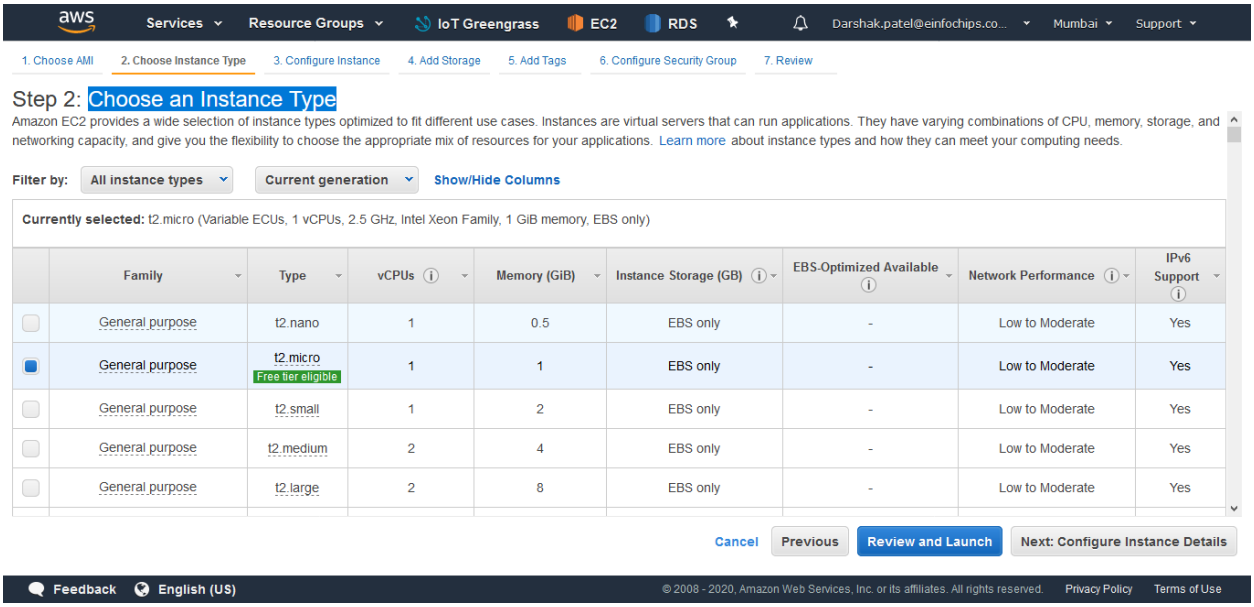


Figure 5: Configure EC2 Instance Type

Step 3: Configure Instance Details (Don't alter anything if don't needed)

- Click on Next: Add Storage

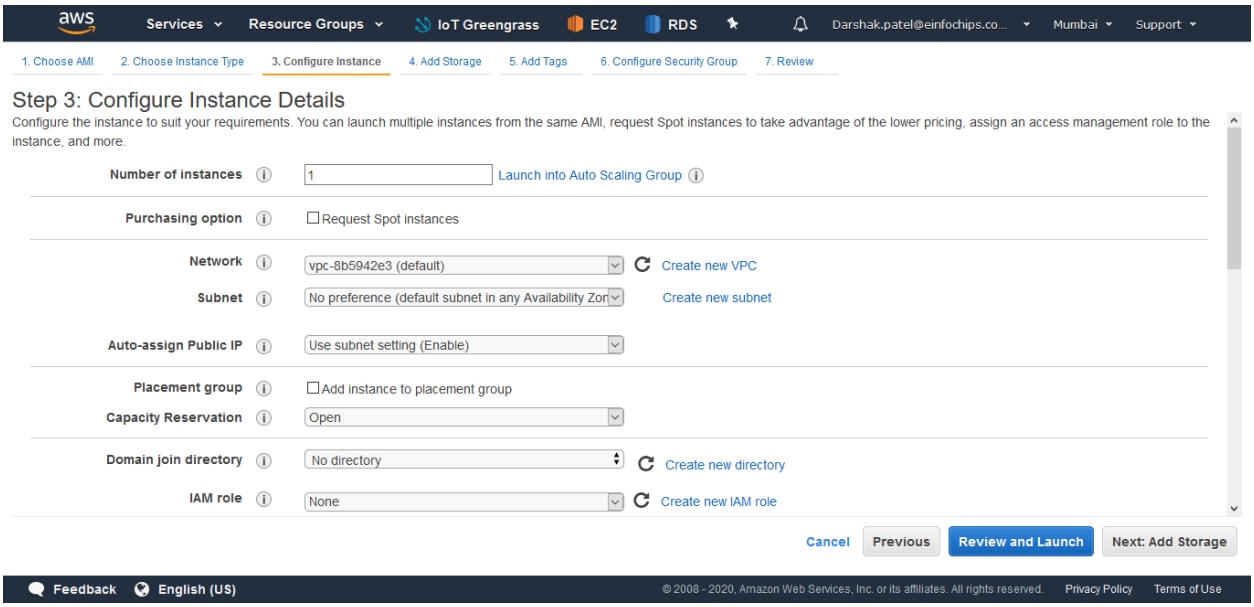


Figure 6: Configure Instance details

Step 4: Add Storage

- Change size to 16 GB (Default 8 GB) then press Next: Add Tags

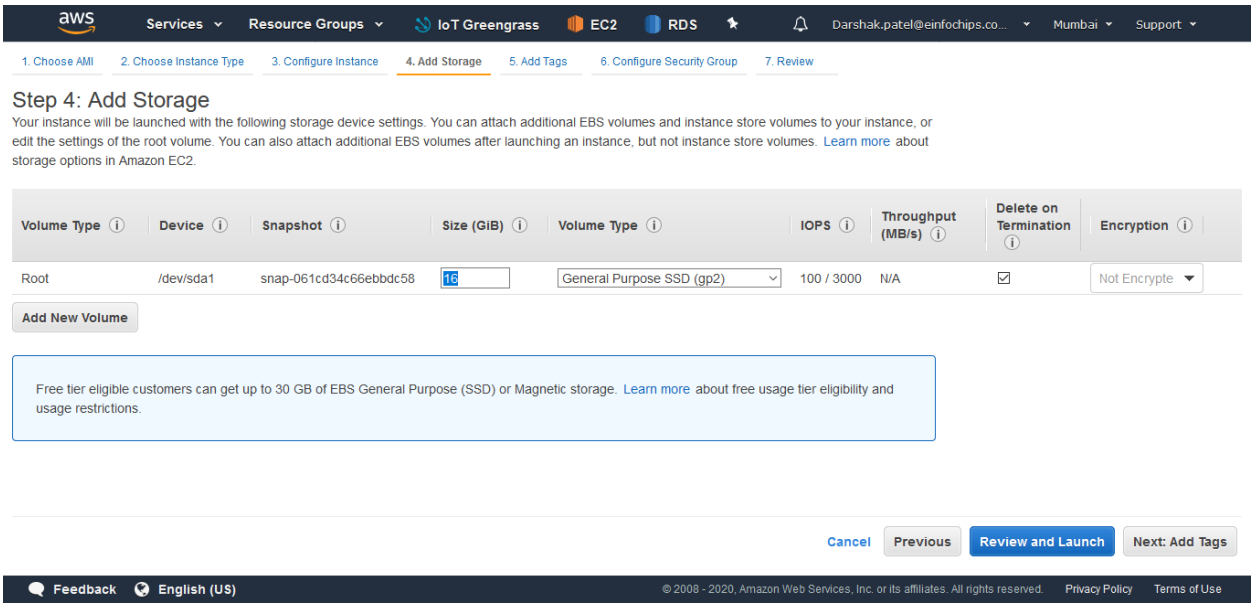


Figure 7: Add Storage

Step 5: Add Tags (Don't do anything)

- Press Next: Configure Security Group

Step 6: Configure Security Group

- Fill up Security group Name: SSK Security Group (Also add description)
- Then press "Review and Launch"

Note: Ensure that both SSH and HTTP are listed as "type" below, otherwise click "Add Rule" to enable those services.

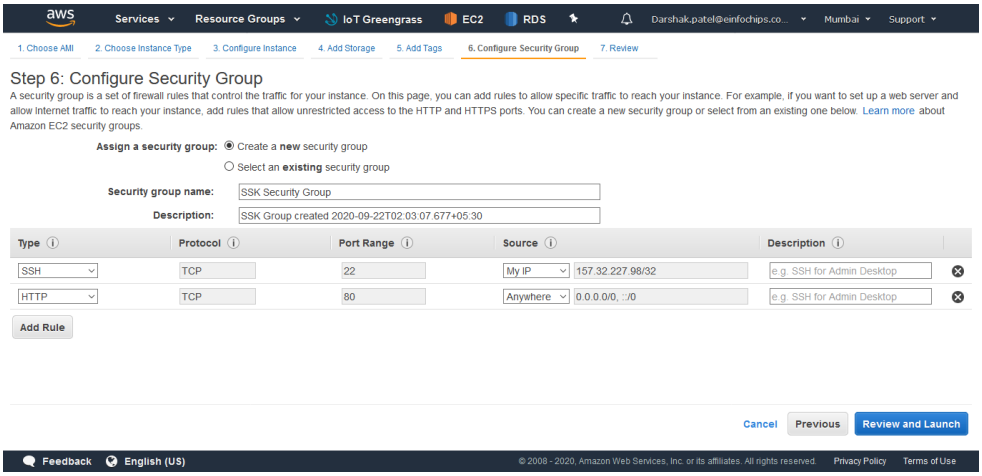


Figure 8: Configure Security group

Step 7: Review Instance Launch

- Press Launch

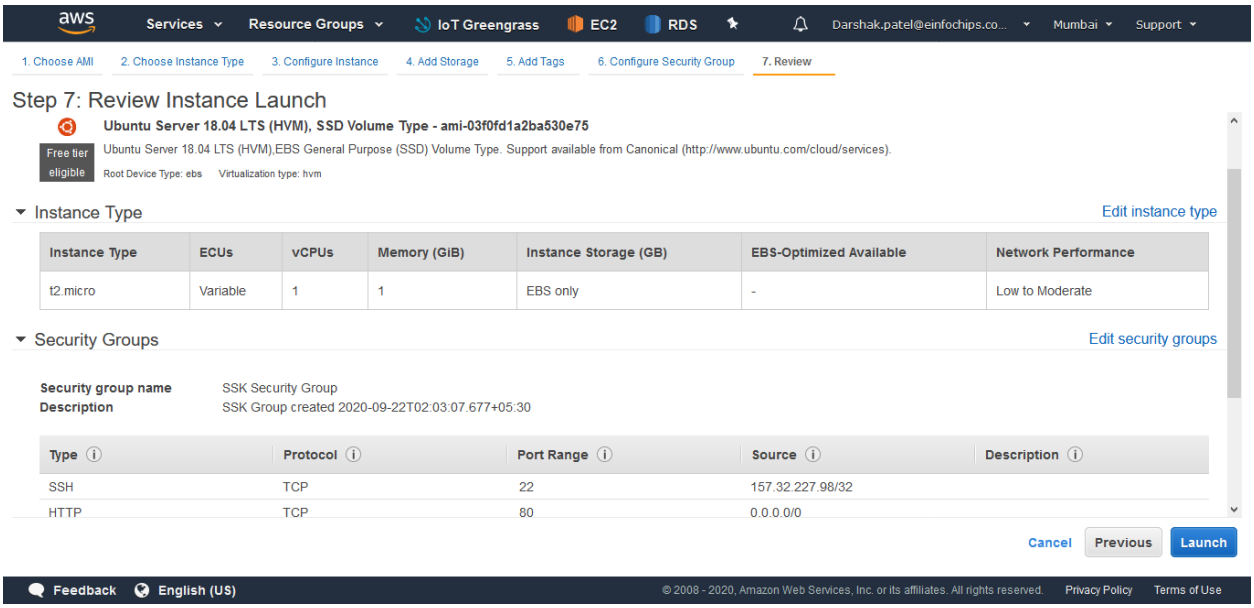


Figure 9: Review Instance

Step 8: Create New key pair

- Select “Create a new key pair” then name “SSK_Key”

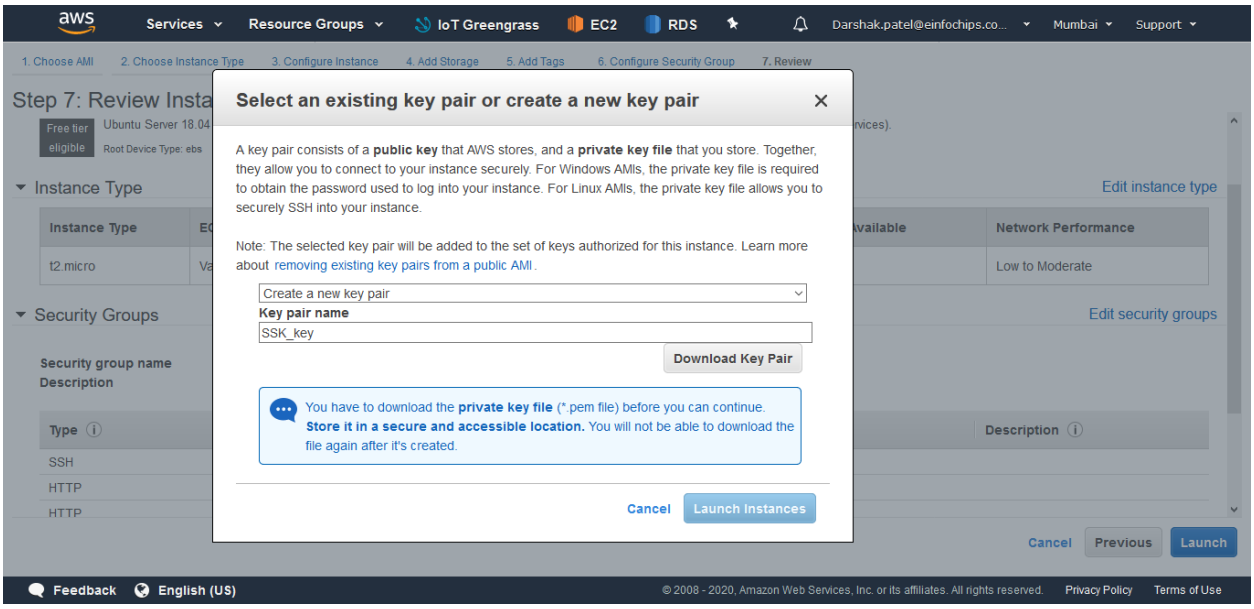


Figure 10: Configure key

Step 9: Download key pair (To Connect EC2 Instance)

- Keep Certificate key file at secure place which will be used to connect EC2 instance.
- Then press “launch Instance”.

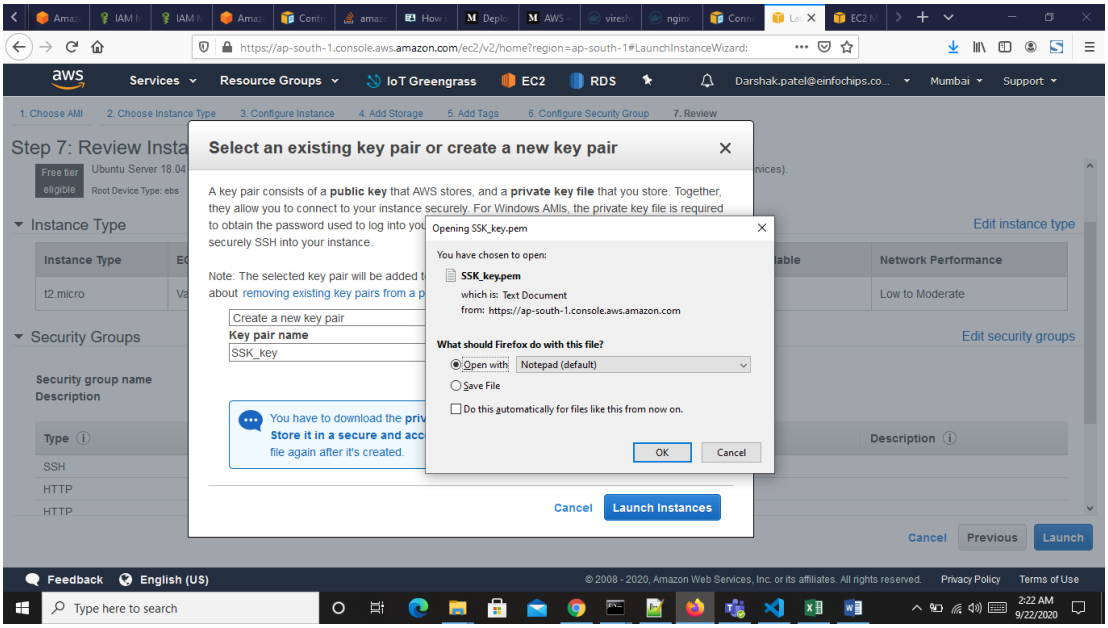


Figure 11: Download key

2.5 Convert key to Putty Format

Step 1: Convert SSK_key.pem file to SSK_key.ppk (Using Putty)

Open PuTTYgen (From Windows) press Load button.

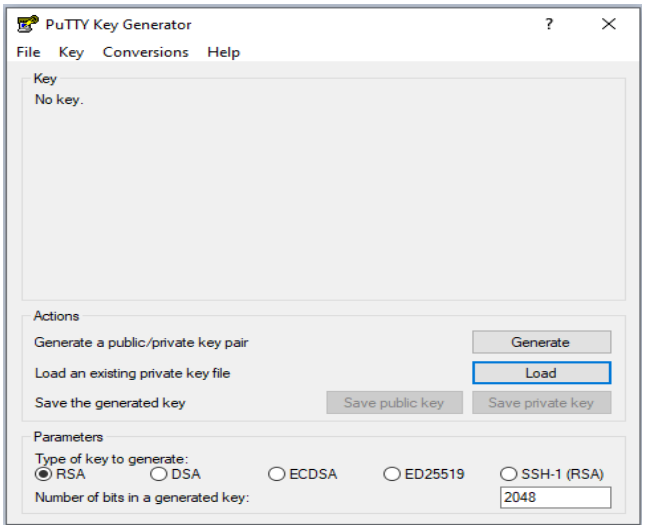


Figure 1: Convert PPK file

- It will ask for file to choose, here you'll need to provide SSK_key.pem file (select all file format)
- After successfull loading of key it will popup the successfully loaded key
- Press "Save private key". (Ignore passphrase warning)
- Name the file " SSK_key" and Save file along with ppk file

Ref Link:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html#putty-private-key

2.6 Configure Putty

- Open Putty and save the session with following details

  Host Name: ubuntu@<host ip address> (Host Ip address can be obtained from EC2 instance)

  i.e.

  Host Name: ubuntu@13.235.8.114

  Session Name: SSK EC2

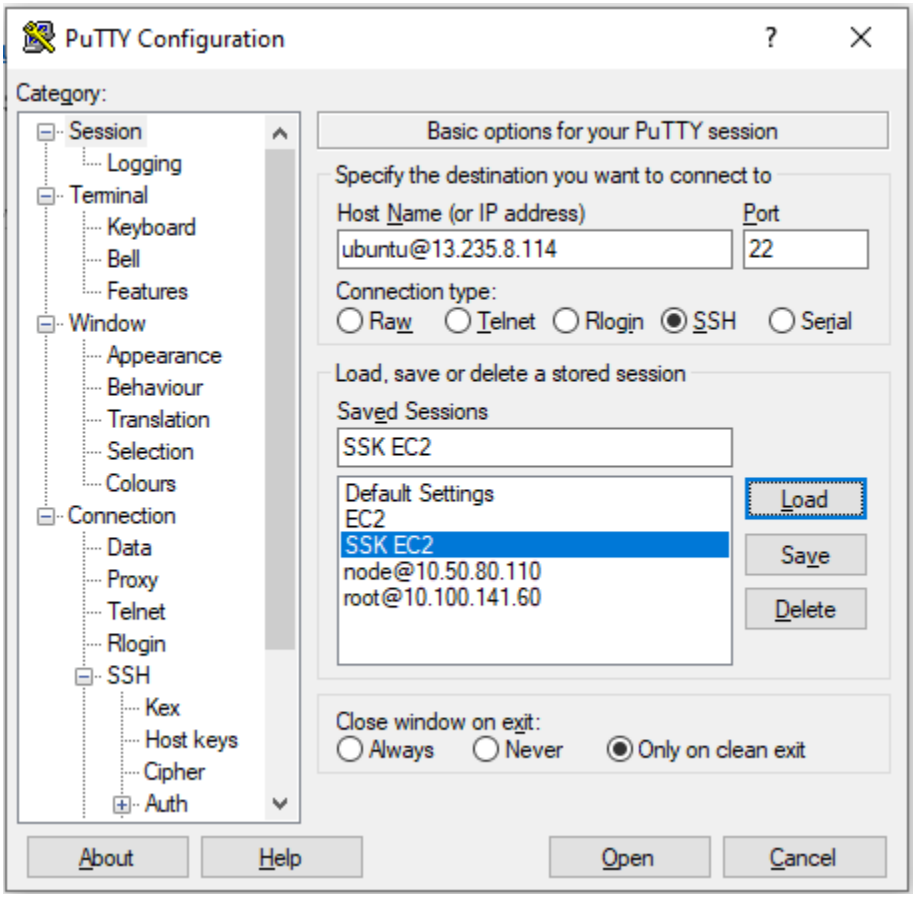


Figure 1: Configure putty

To configure key

- Go to Connection >> SSH >> Auth >> Select private key
- Then again save it and press open button.

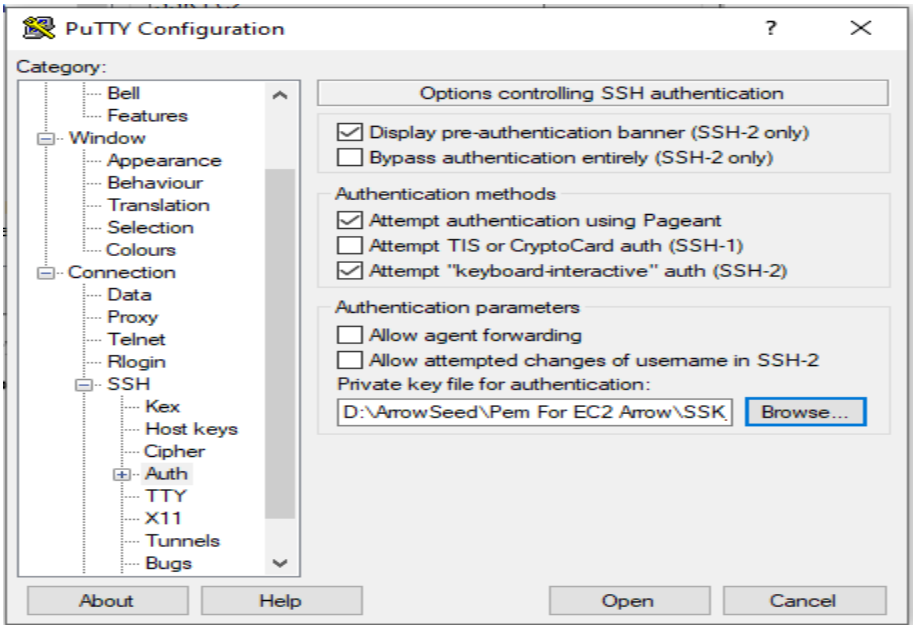


Figure 2: add key to putty

- Here you can now connect to the AWS EC2 Instance.

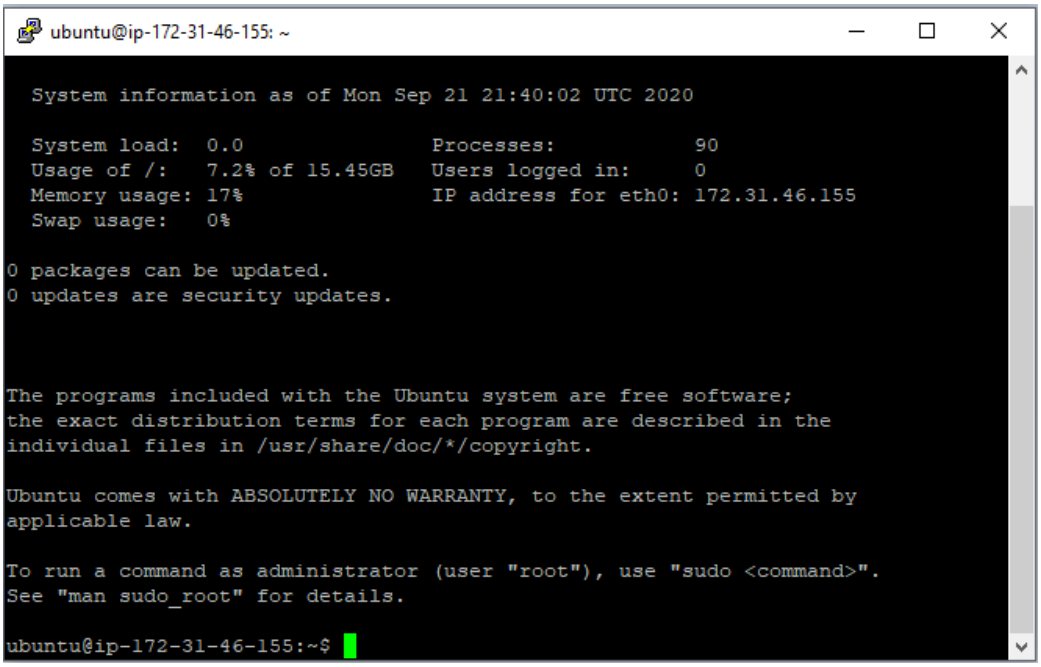


Figure 3: Connected AWS

3 INSTALLING DOCKER ON EC2

3.1 Execute below command

Step 1: Update your existing list of packages

$ sudo apt-get update

Step 2: Next, install a few prerequisite packages which will let apt use packages over HTTPS:

$ sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common

Step 3: Add Docker's official GPG key:

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

Step 4: Add the Docker repository to APT sources

$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

Step 5: Update the package database with the Docker packages

$ sudo apt-get update

Step 6: Install Docker

$ sudo apt-get install docker-ce docker-ce-cli containerd.io

Step 7: To verify installation

$sudo docker –version

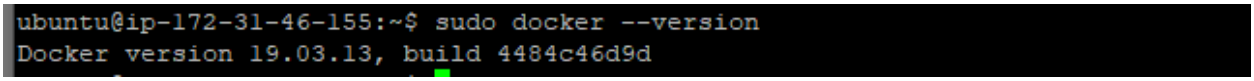


Figure 1: Docker version

4 CONFIGURATION OF EC2 INSTANCE, RDS SERVICE AND SQL DATABASE

4.1 Application access

After executing docker run command to access application using following page:

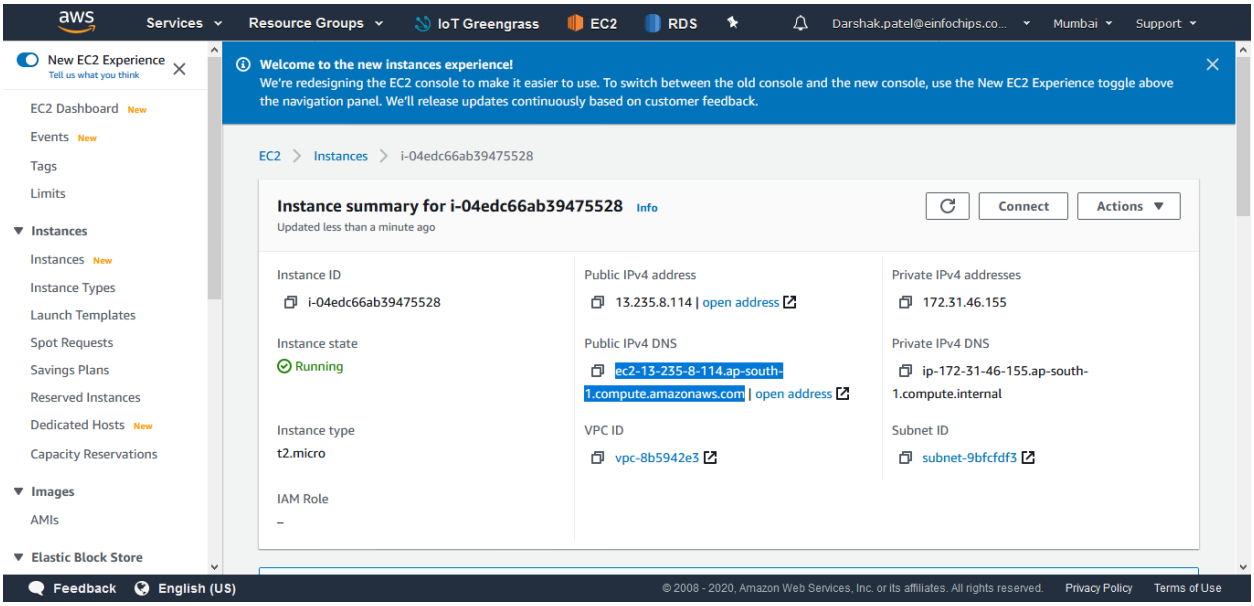


Figure 1: Application access

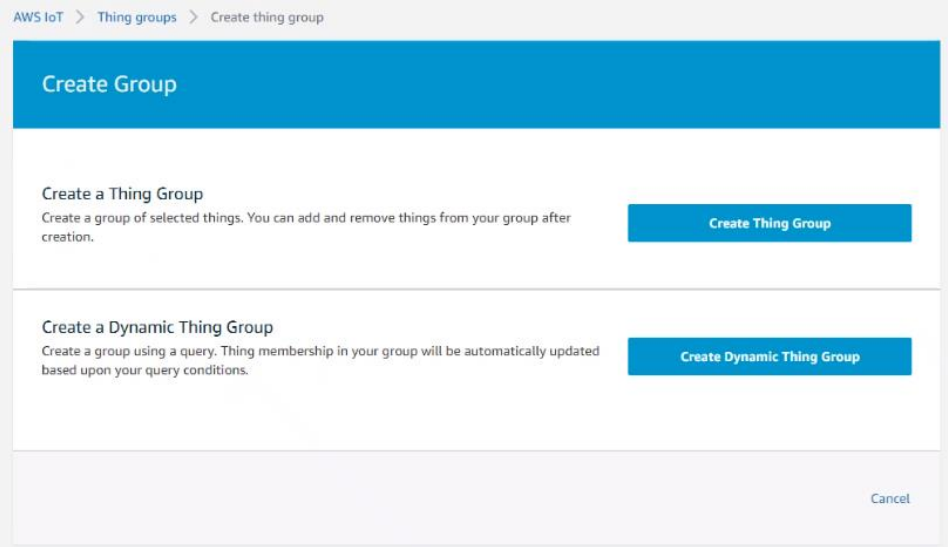
Make note of the Public IPv4 DNS address provided and it needs to be in the following format, with a leading HTTP:// as shown below;

Public Access URL: http://ec2-13-235-8-114.ap-south-1.compute.amazonaws.com/

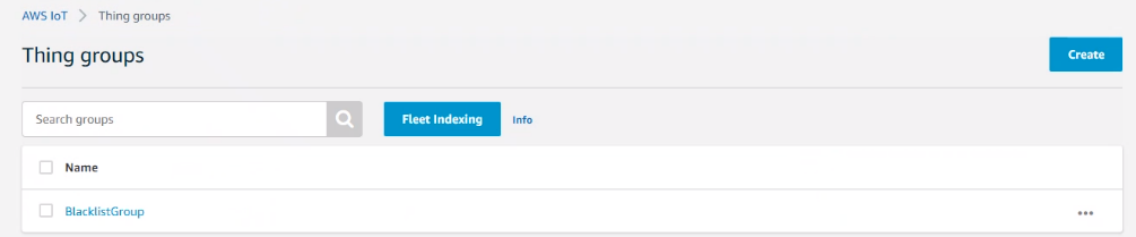
4.2 Application – "Allow" or "Deny" listing

1. Create Thing group "BlacklistGroup" and create one default policy "blacklist-policy" then attach policy to thing group.

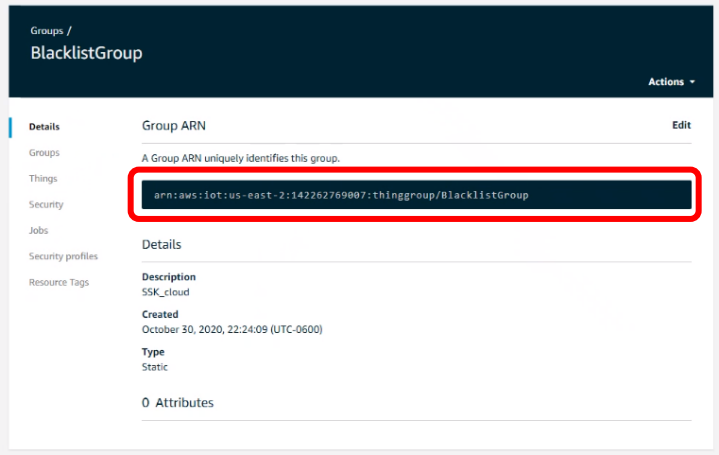
    - To create 'Thing group' : navigate to IoT Core → Manage → Thing groups → Create



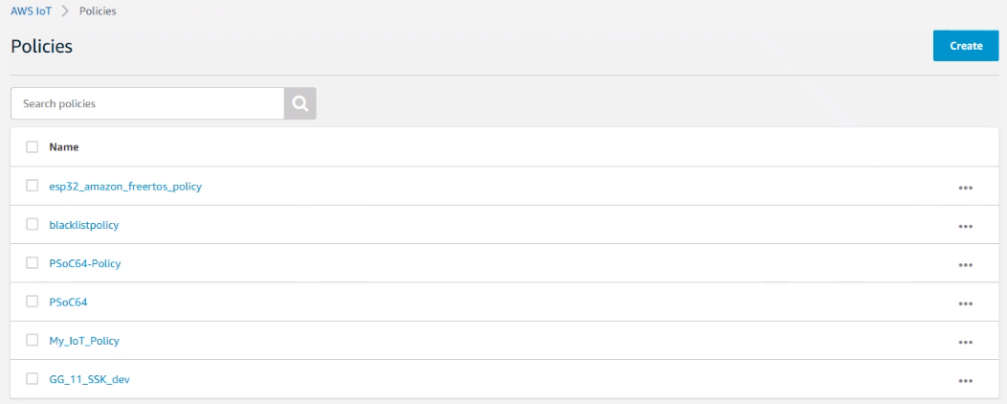
    - Provide a name to your Thing Group, like what is shown below.



    - Click on "BlacklistGroup" (or whatever name you gave it) and make note of your Group ARN listed below;



2. To create default policy: Navigate to: IoT Core → Secure → Policies



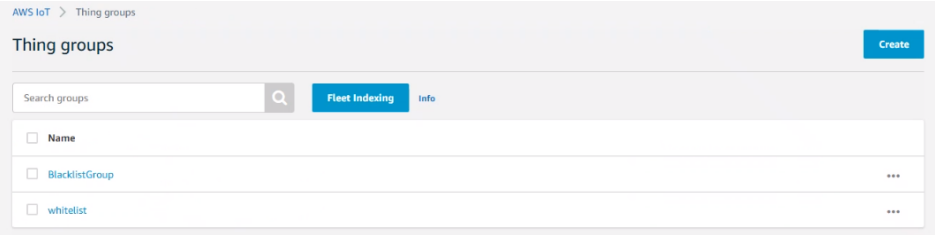
- Click create and enter the name of your policy. Under Add Statements click Advanced mode. JSON statement will be seen and then edit with the information as shown below.

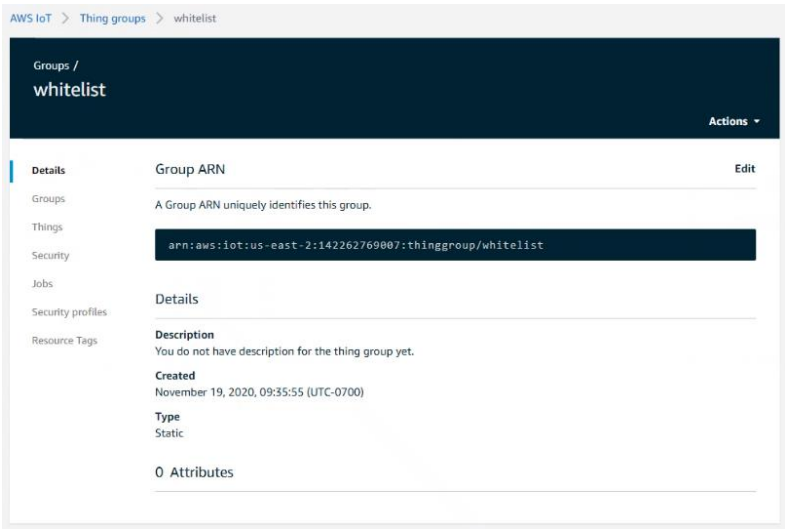
  Note: You will be entering your specific ARN Group provided in the previous step next to "Resource";

```
{
        "Version": "2012-10-17",
        "Statement": [
        {
          "Effect": "Deny",
          "Action": "iot:*",
          "Resource":  "Group ARN noted from the step above:topic/replaceWithATopic"
        }
      ]
}
```

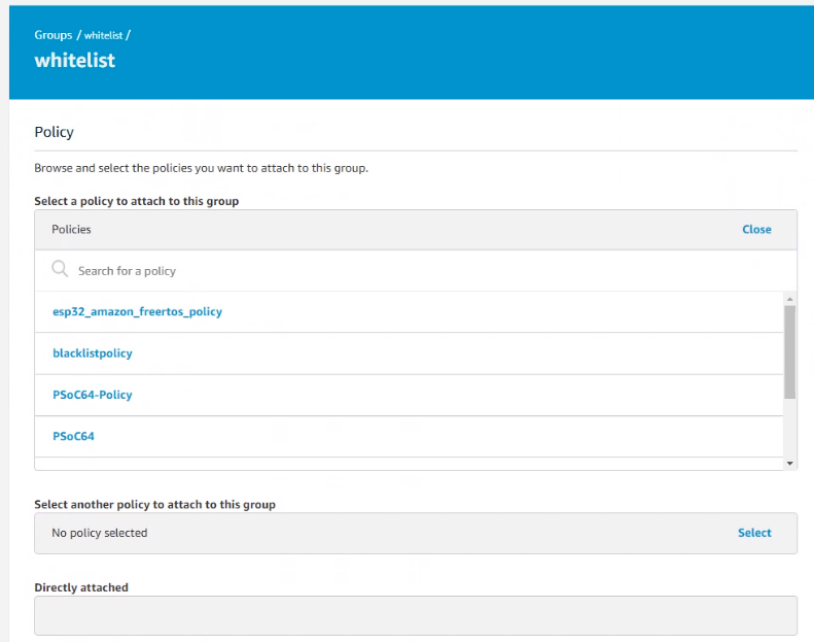
3. Next, you need to attach the "Policy" to the "Thing Group";

   - Navigate to: IoT Core → Manage → Thing Groups and click on the Group you just created;

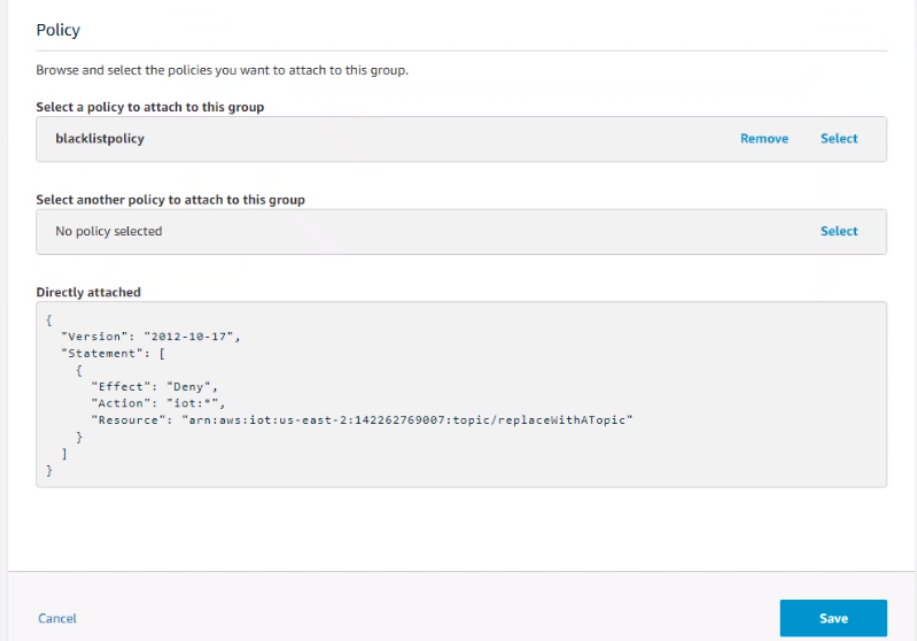


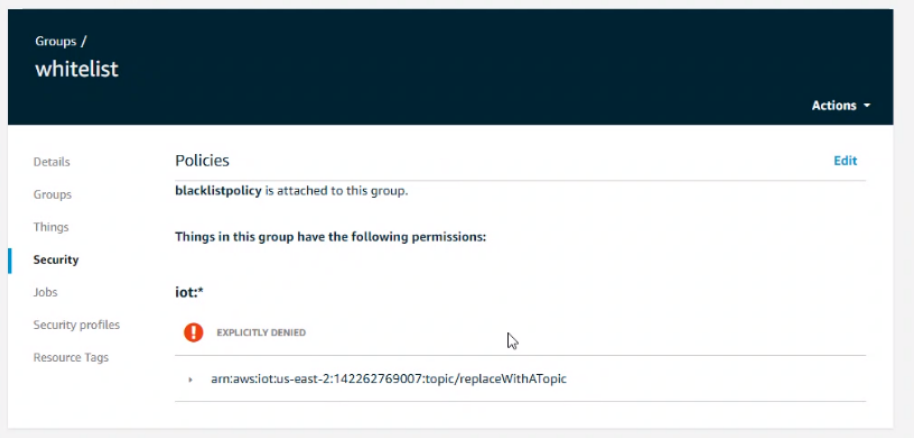


- Click “Security” on the left and then “Edit”, Select the “Policy” you recently created;



- You should see the policy statements you had edited from the previous steps, then click "Save"





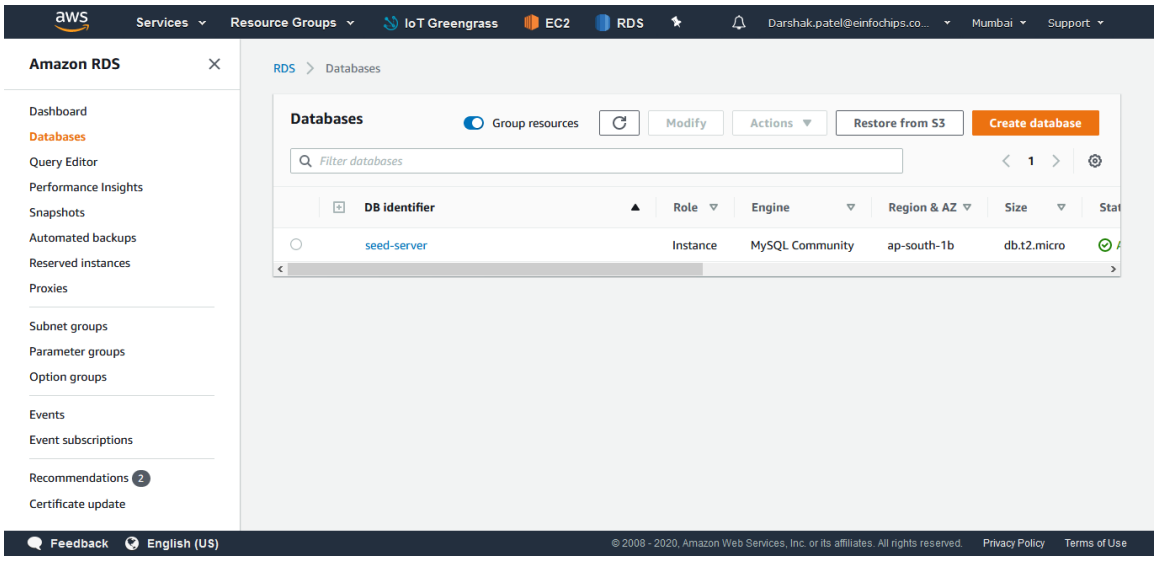
4. In order to perform OTA updates, the user will need to Create an OTA Role and the link to the instructions within AWS is provided below. You will also need to create an OTA Job, which is part of the SSK Cloud Connect Tool and outlined in Section 5 of the SSK Cloud Connect Users Guide;

   To create OTA update role follow below URL:

   URL: https://docs.aws.amazon.com/freertos/latest/userguide/create-service-role.html

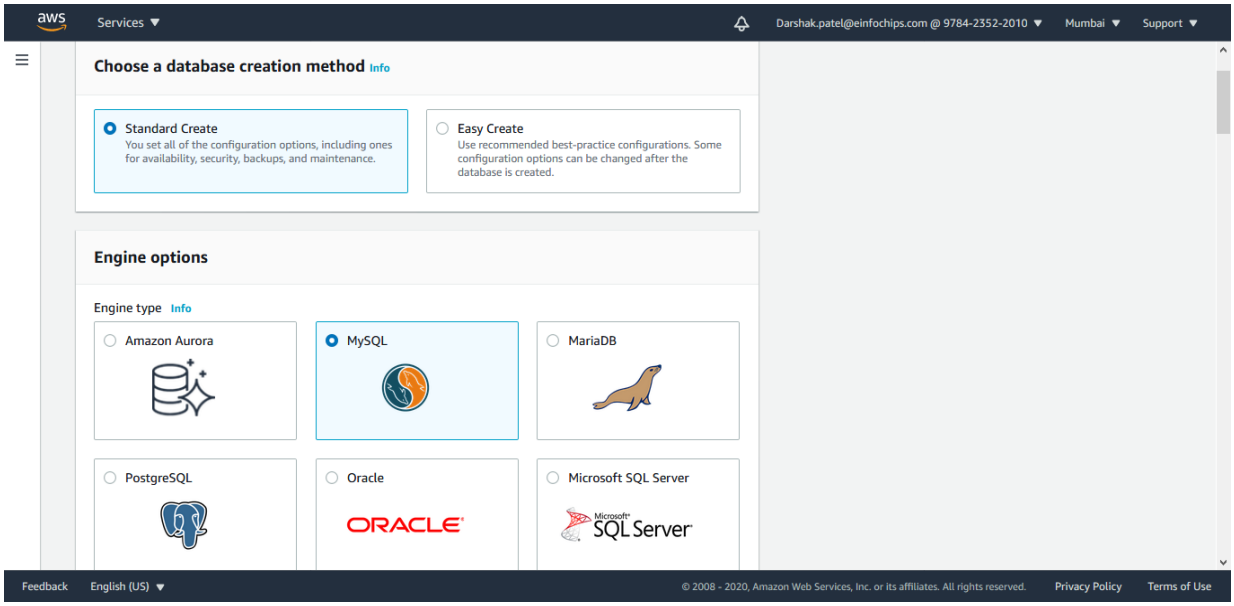
4.3 AWS RDS Service – Database Setup and Configuration

- Go to Services >> Database >> RDS(Select)
- Click Left side navigation “Databases” will show following page.

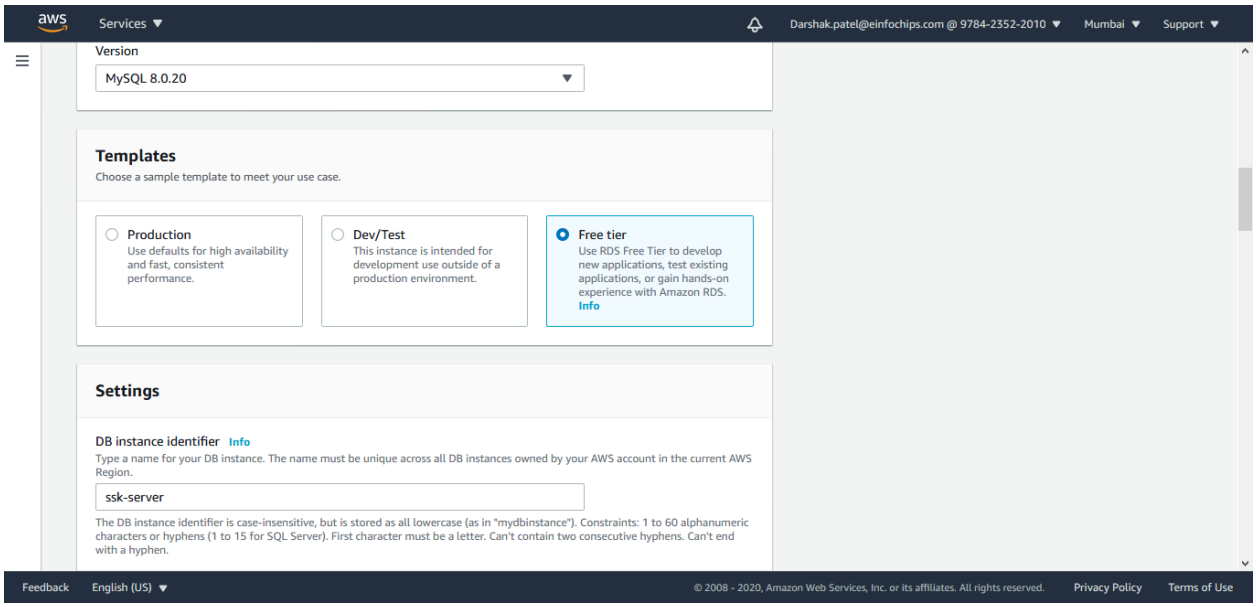


4.4 Creating a Database

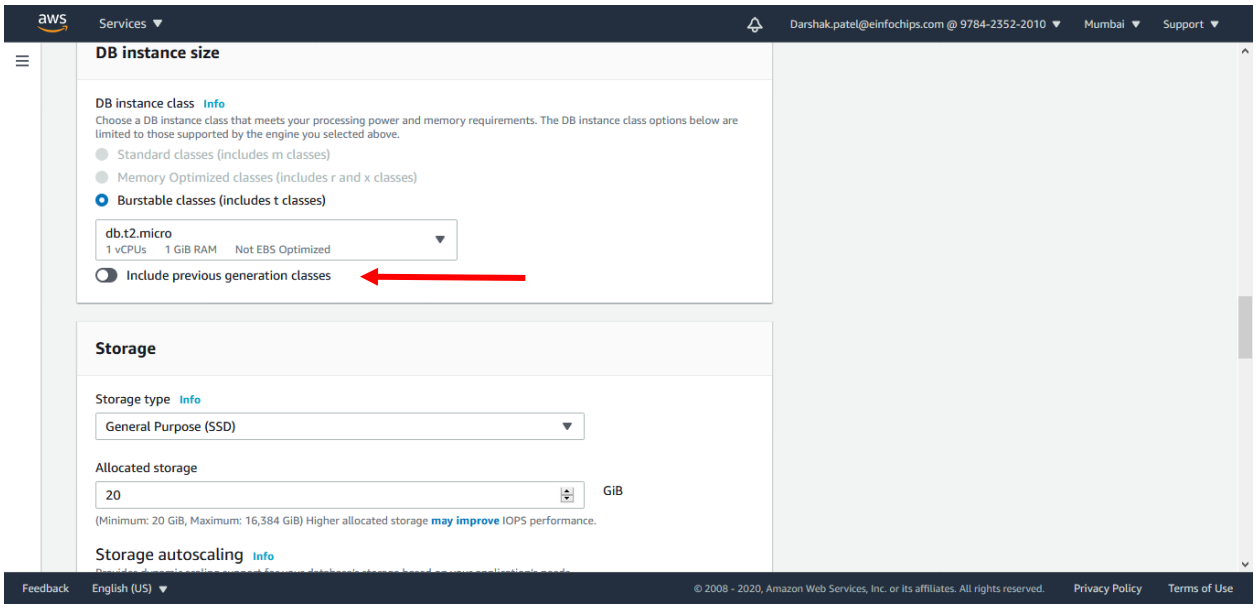
- Select “Standard Create”



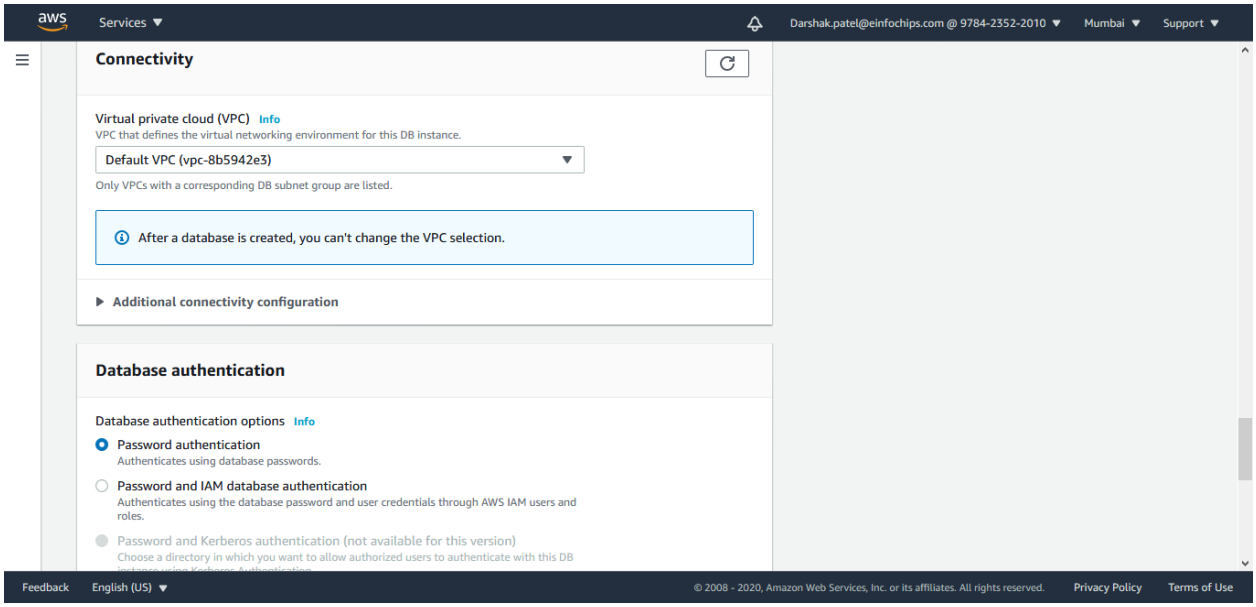
- Select “Free Tier”



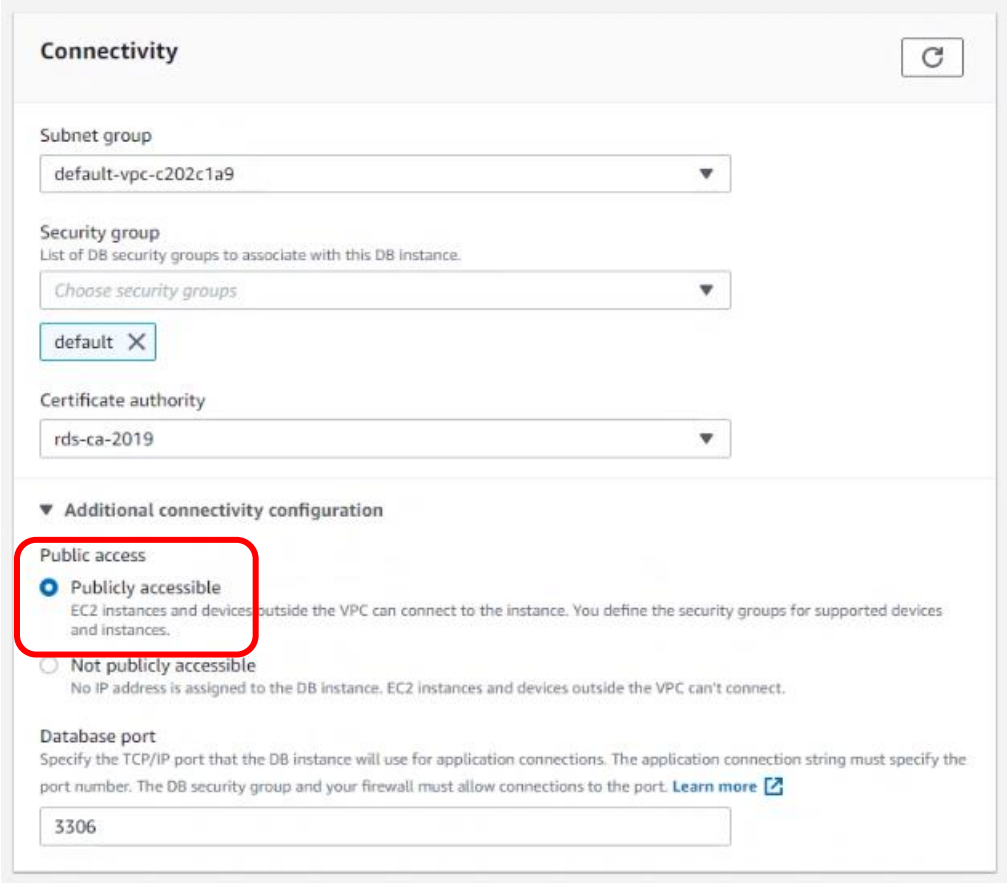
- Enable “Include previous generation classes”



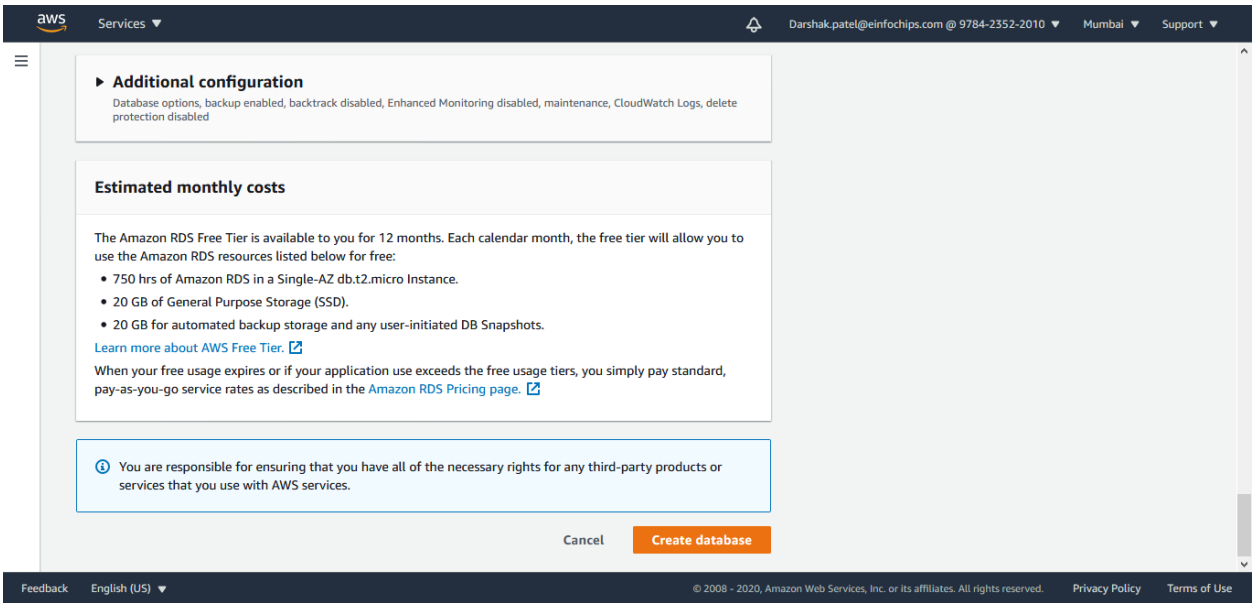
- Select “Default VPC” for the Virtual Private Cloud and “Password Authentication”
- Make note of the database password entered

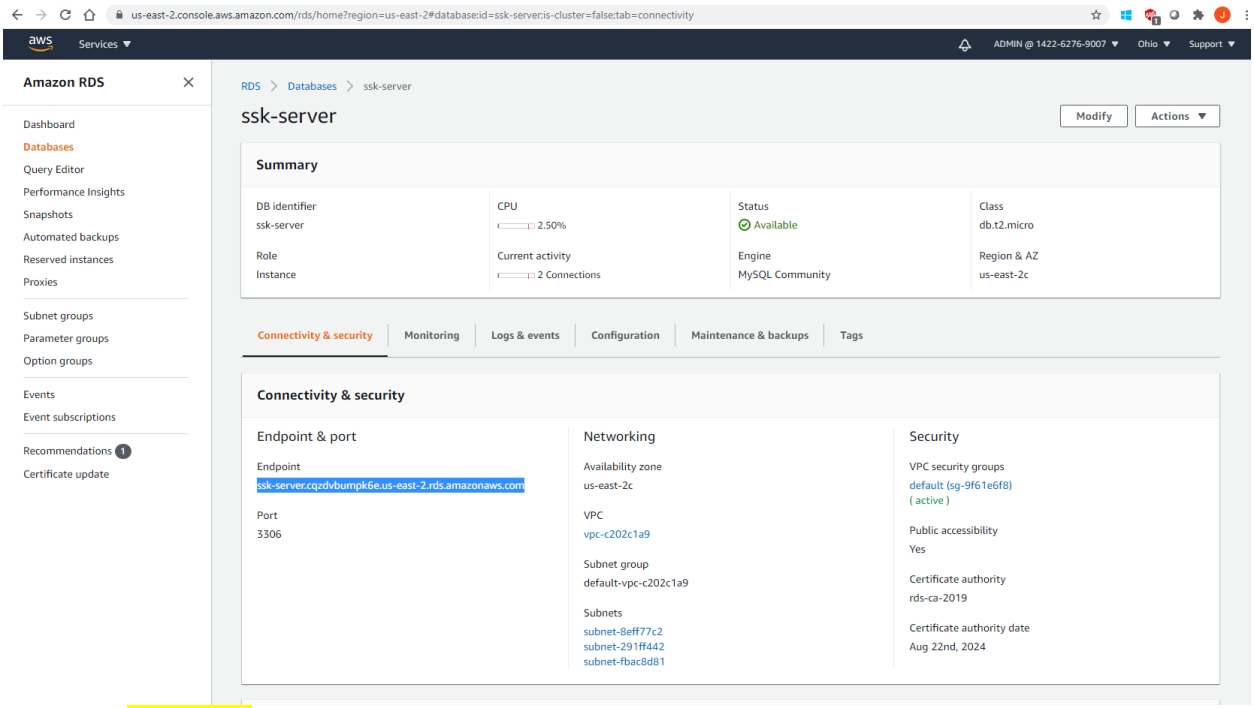


- Select “Publicly accessible” under Additional Connectivity Configuration

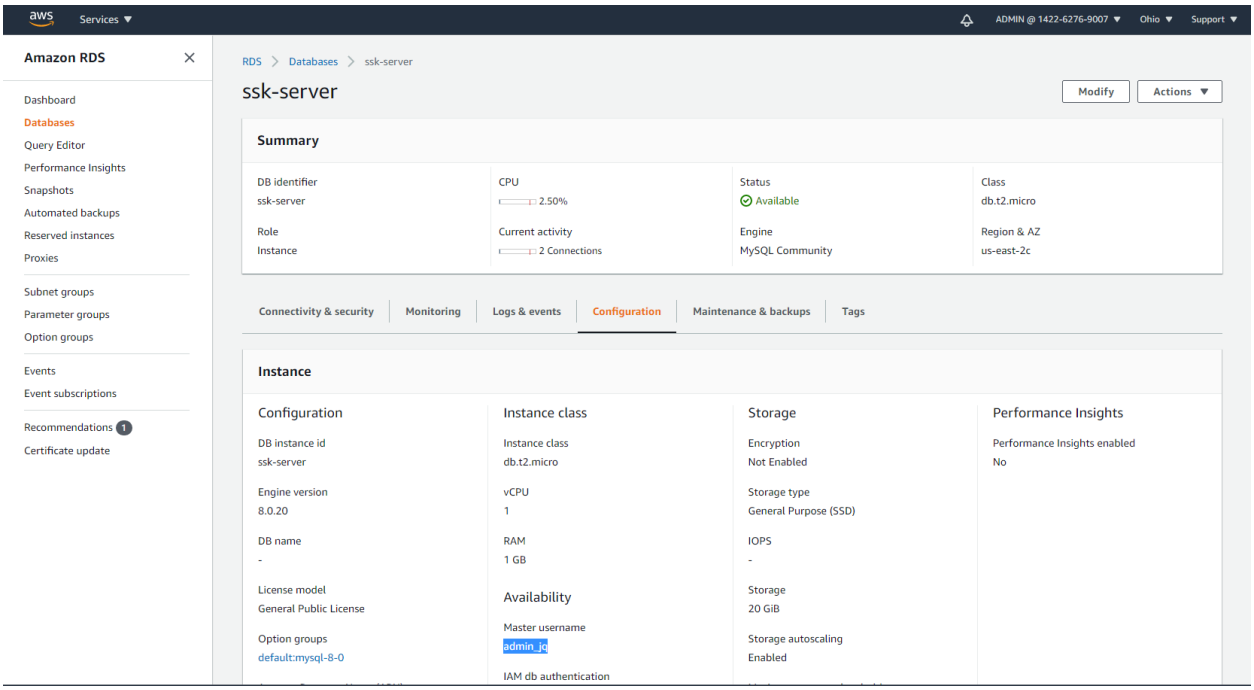


- Click “Create database”



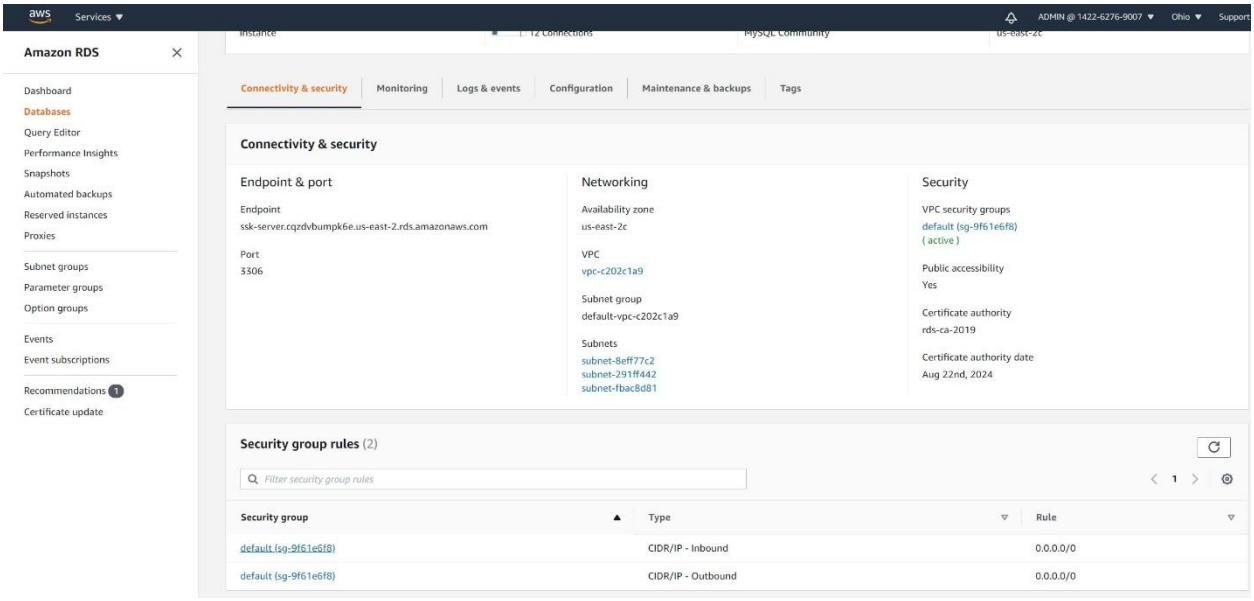


- Make note of the RDS URL that is created and highlighted above.

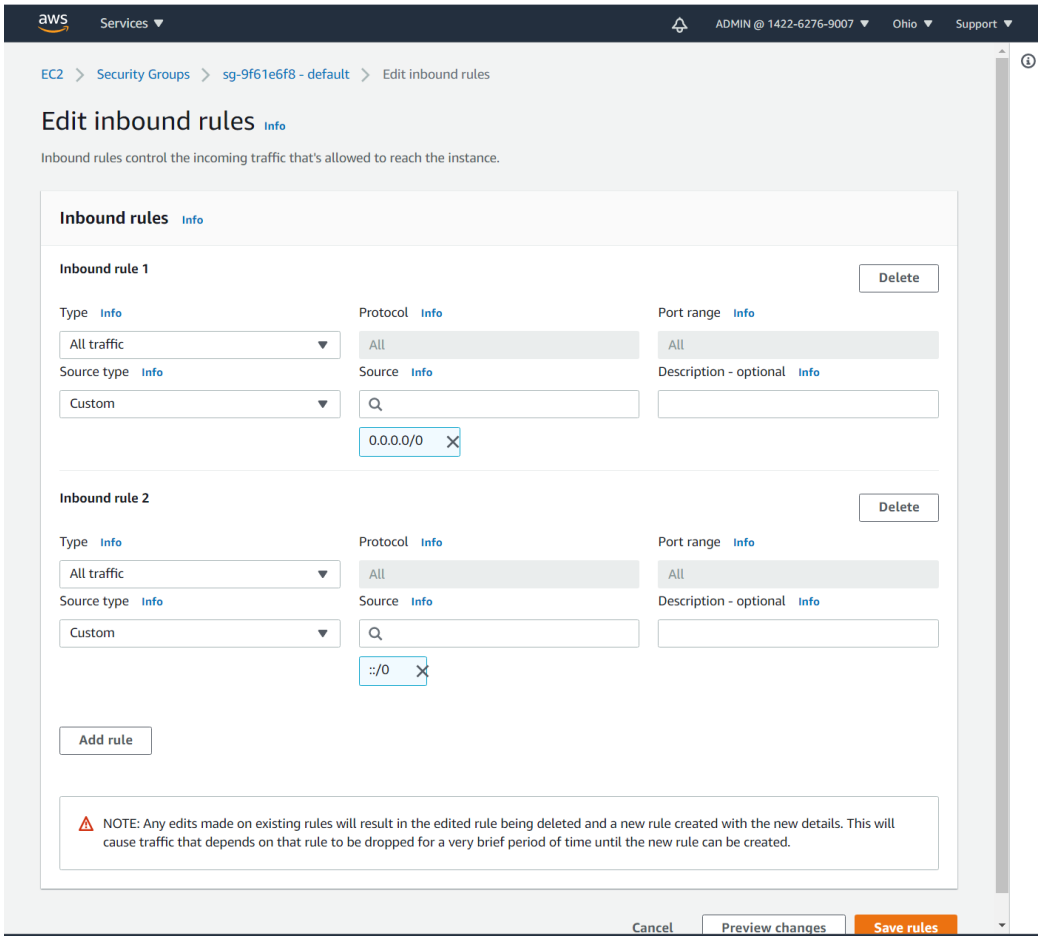


- Make note of the User Name highlighted above, under the configurations tab.

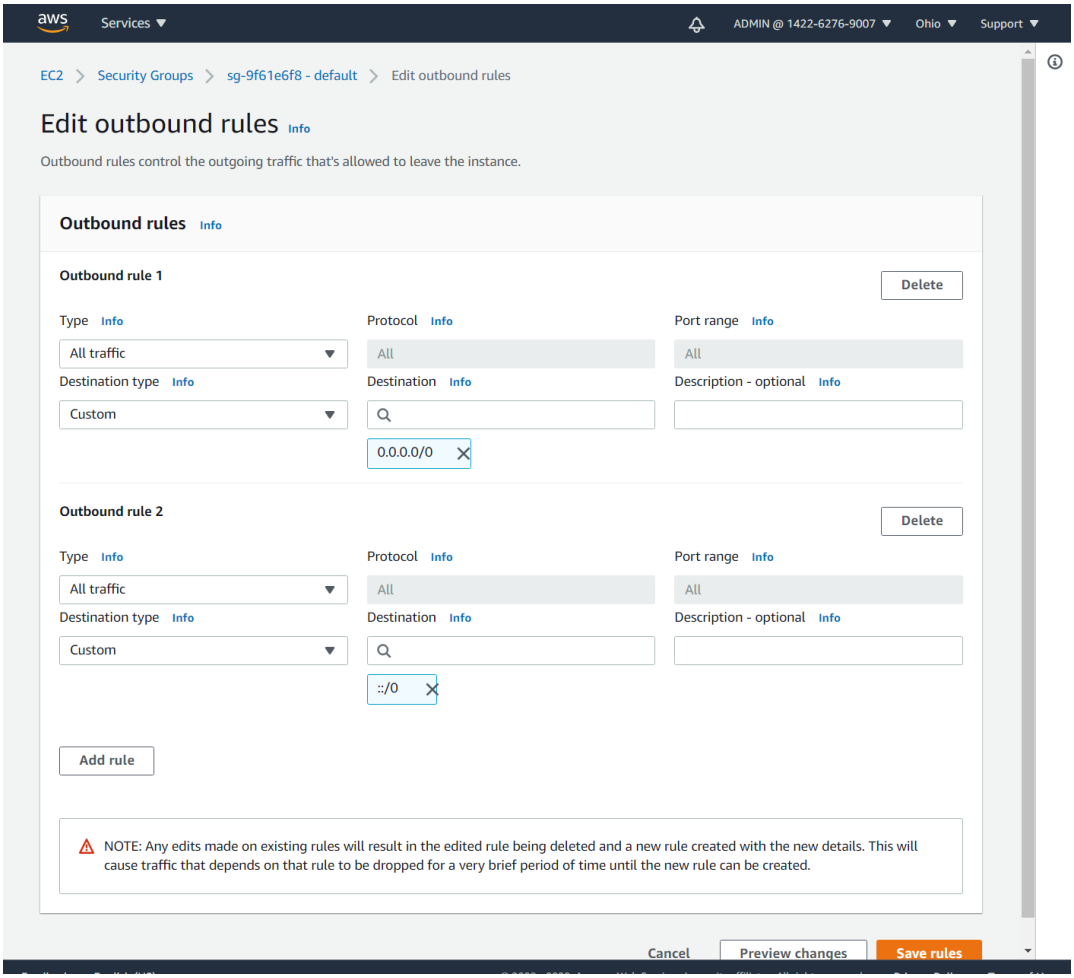
- Once MySQL Database is created, ensure the below Security group rules are set by clicking on the default security group under “Security group rules” in Amazon RDS



Inbound Rules:

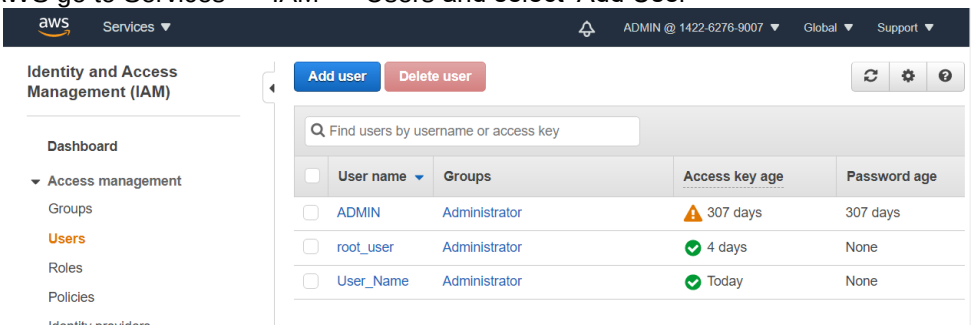


Outbound Rules:

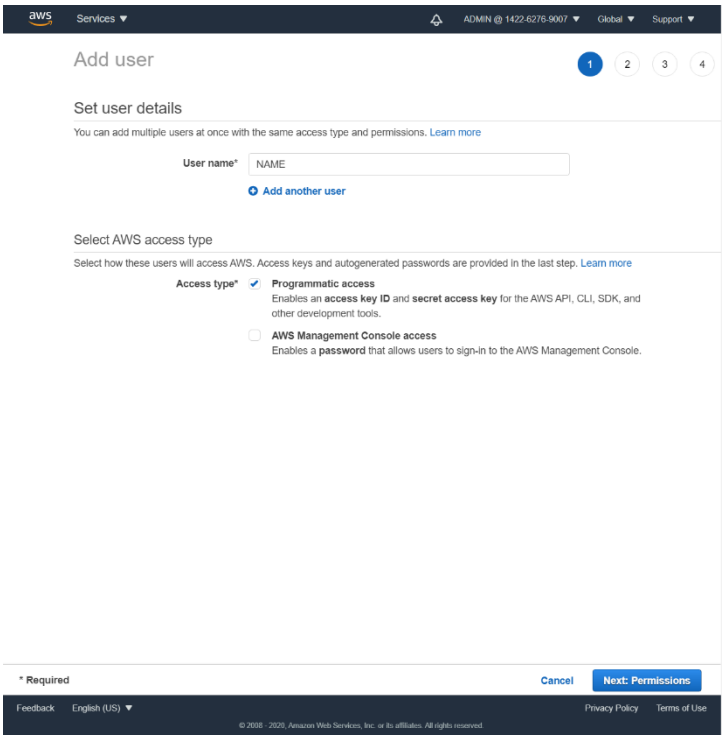


4.5 Creating an IAM User

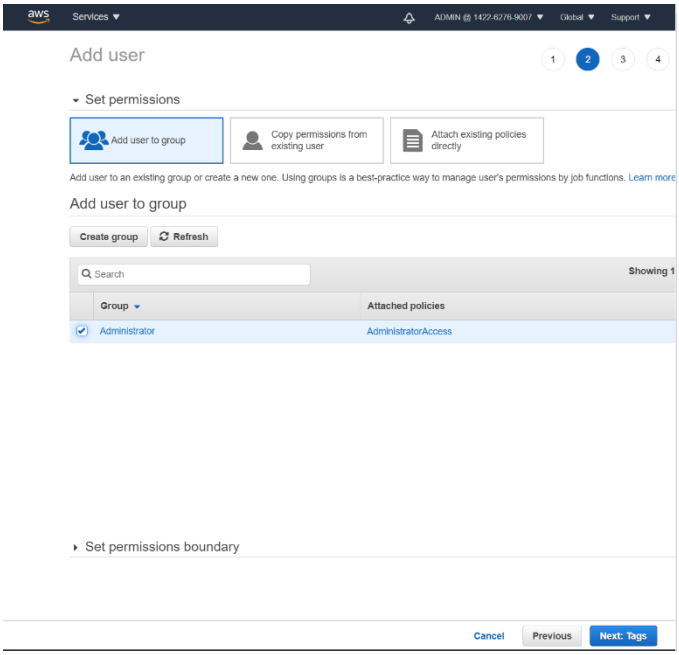
- In AWS go to Services >> IAM >> Users and select 'Add User'



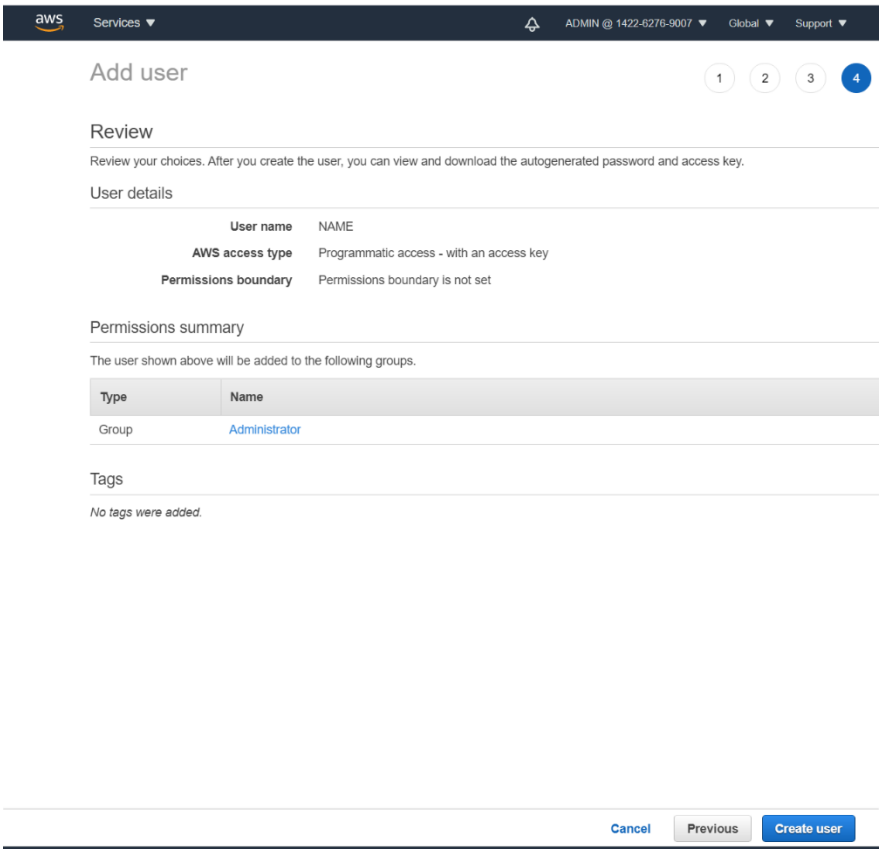
- Pick a username and give it Programmatic access. Note this will be the username you will use to log into SSK Cloud Connect



- Choose 'Next: Permissions' and add your IAM user to the Administrator group with the Administrator Access policy. If you don't have an Administrator group then choose "Attach existing policies directly, search for the 'Administrator Access' policy and attach it



- Click Next: Tags >> Next: Review >> Create user



- Download the 'new_user_credentials.csv' and save it in a safe location

4.6 MySQL Setup and Configuration

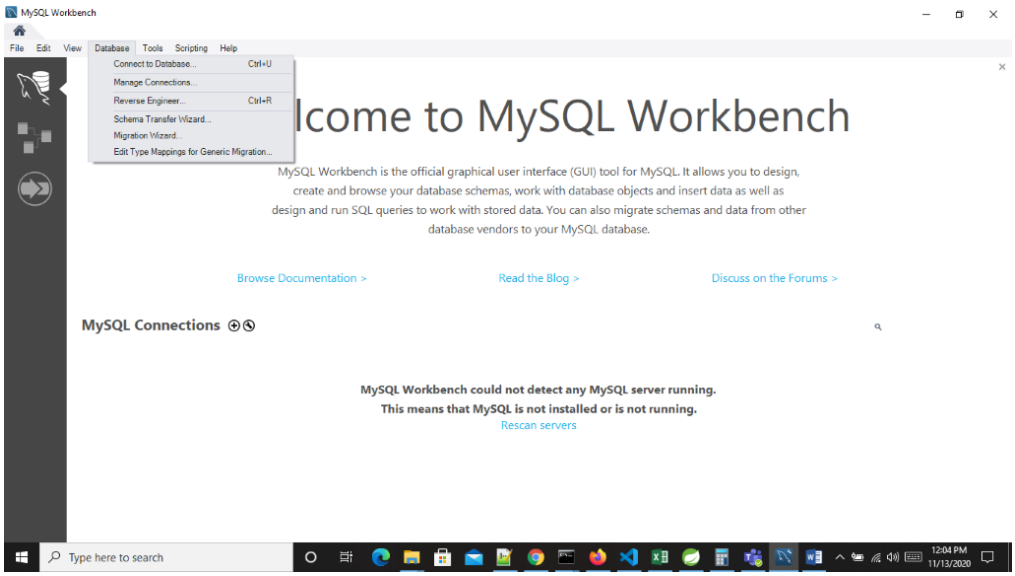
1. Install MySQL Workbench (link provided below), then Open MySQL workbench.

   https://dev.mysql.com/downloads/workbench/

2. Install Postman

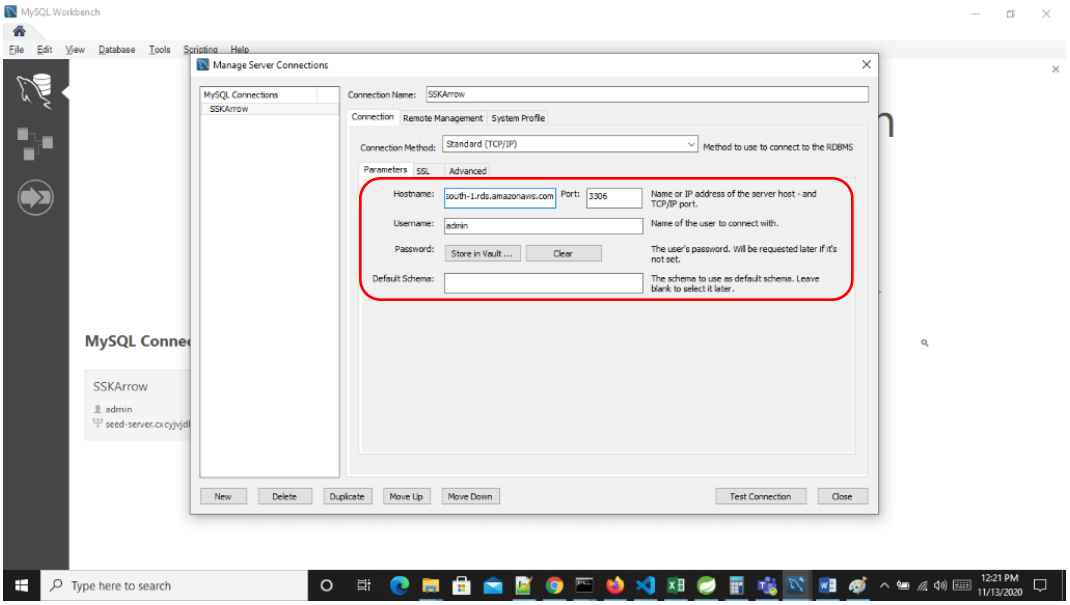
   https://www.postman.com/

3. On Tab Database & select Manage connections.(Database -> Manage Connections)

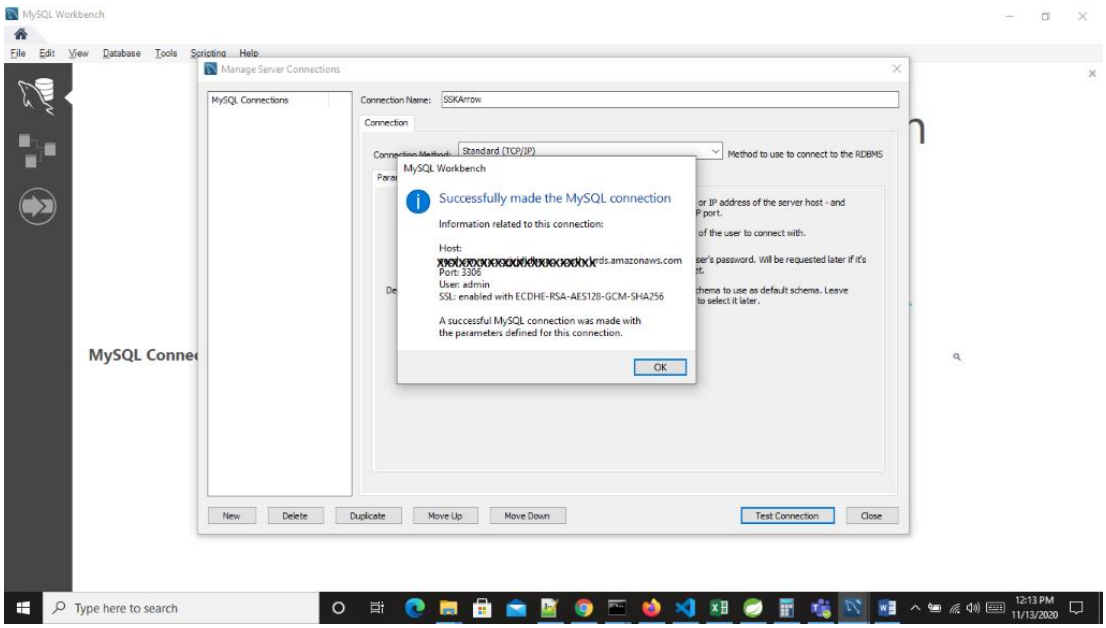


4. It will open pop up model. Press New Button then fill up details as per below:

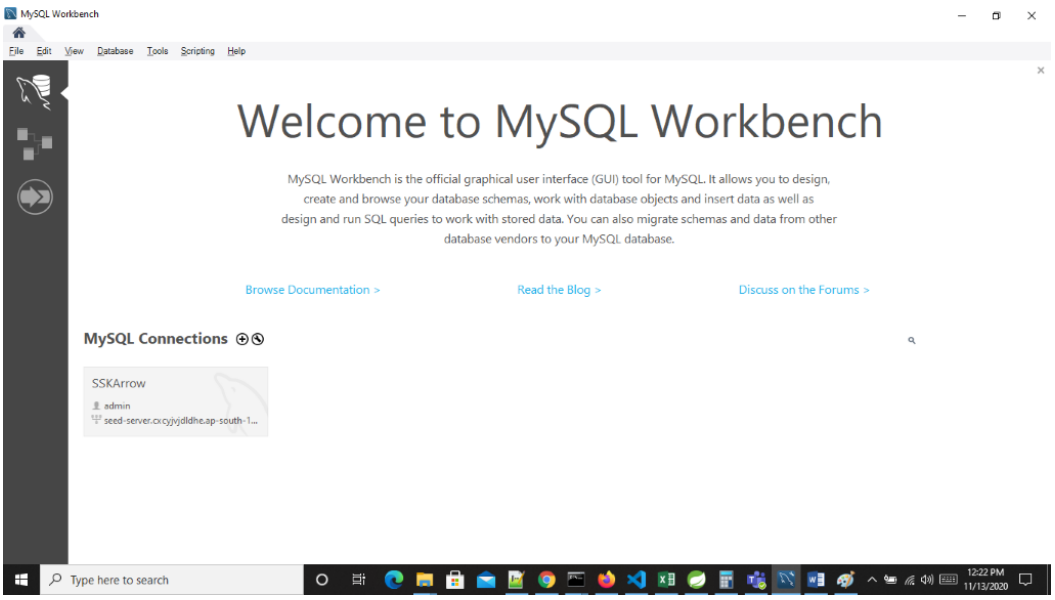
   Connection Name: <Name of connection>
   Host Name: <AWS RDS HOST URL>
   Port: 3306 (Default value)
   User Name: < AWS RDS User name>
   Password: <AWS RDS user password> (Store in Vault if needed)



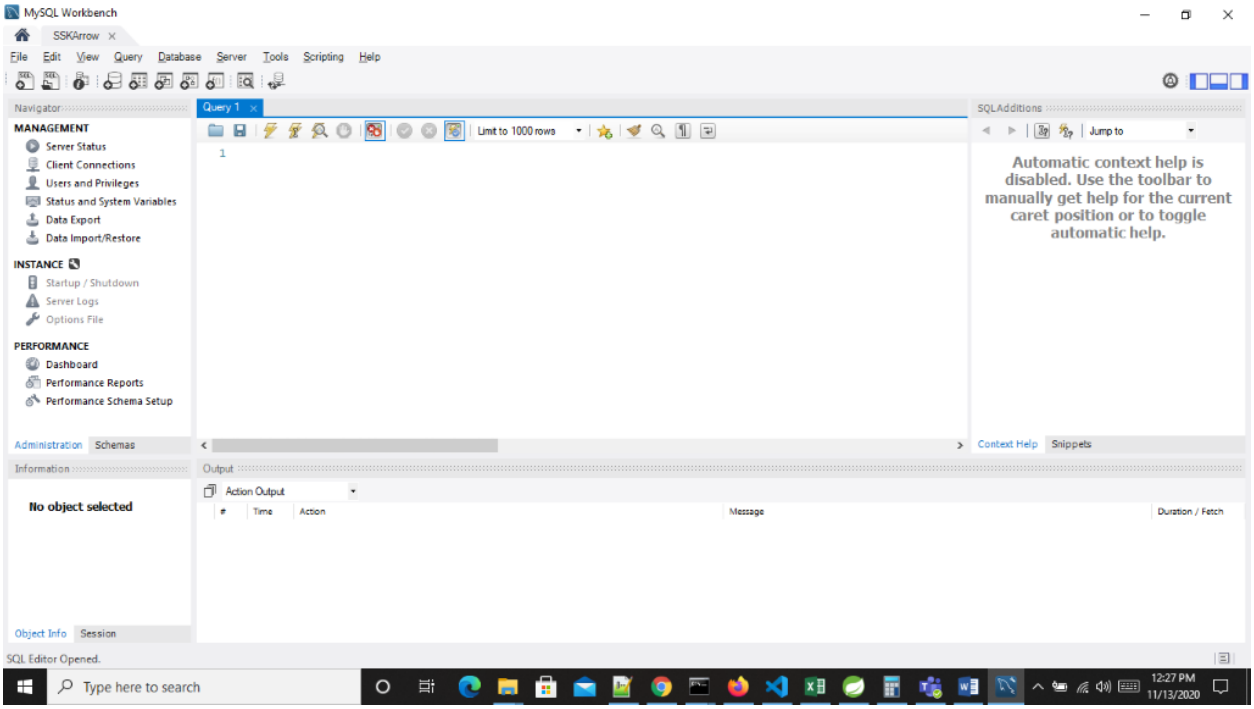
5. Press on Test Connection. It will pop up successful connection message. Then click on close button



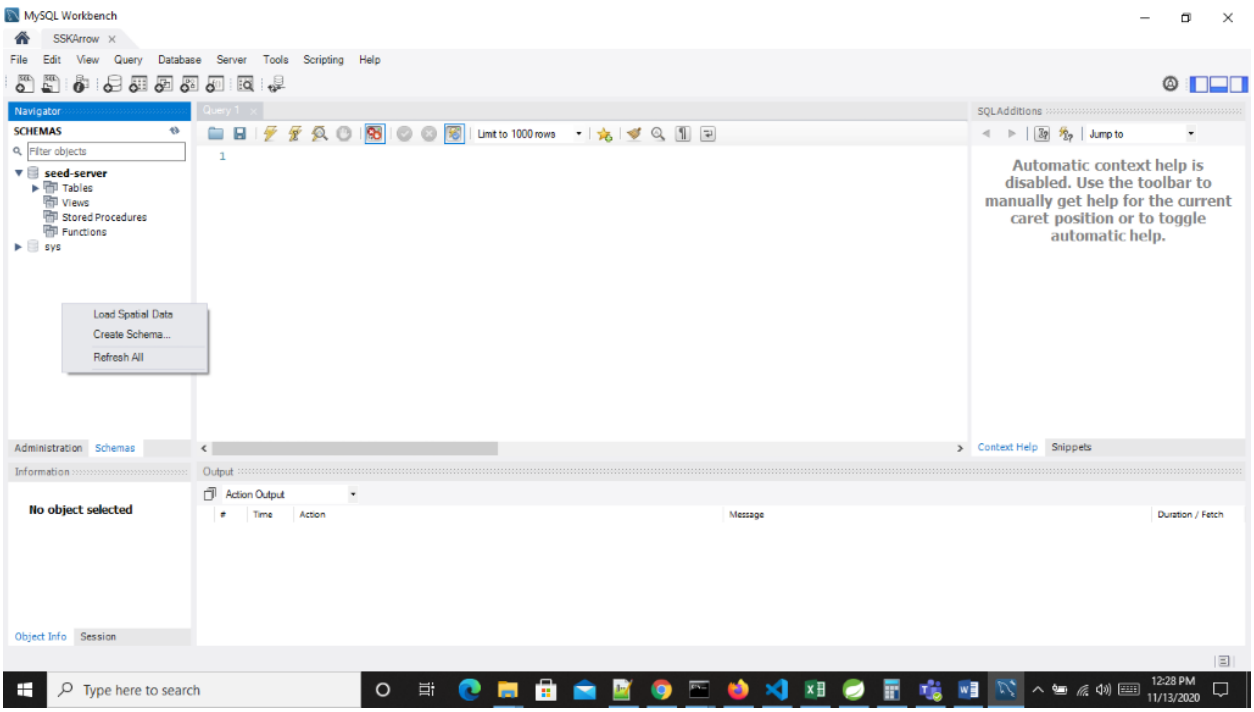
6. Further it will show following details. Click on created connection button.



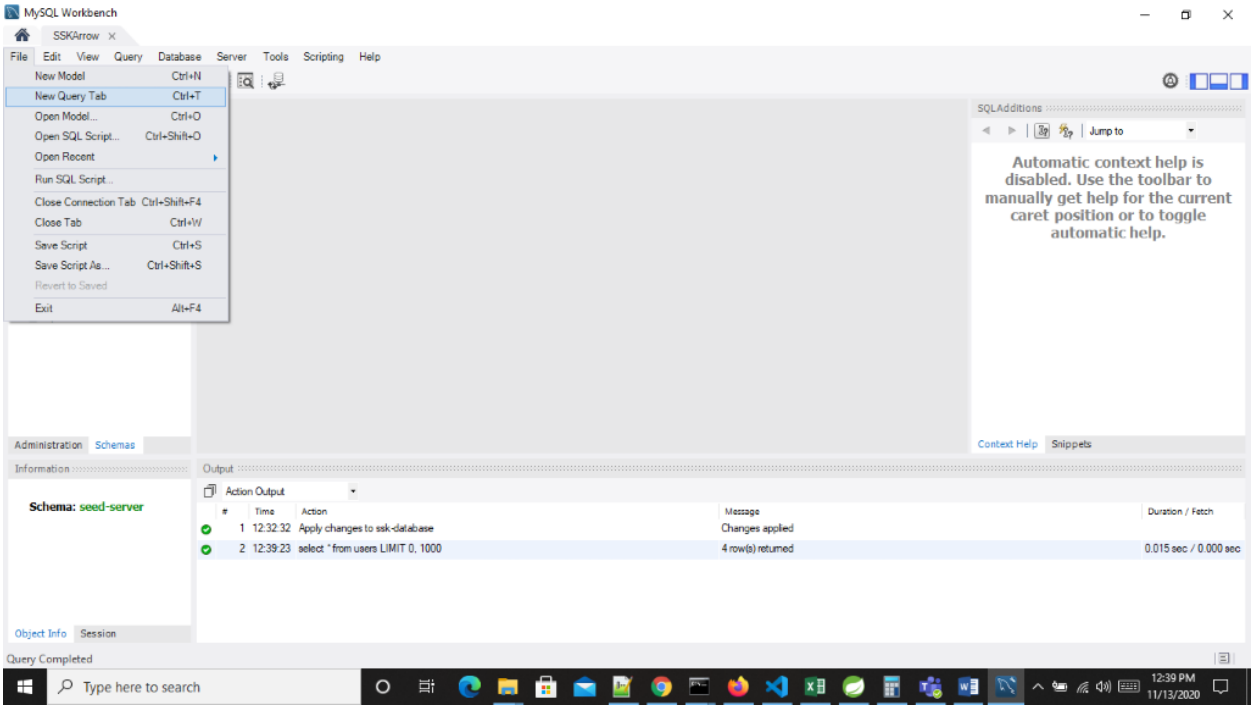
7. It will open Administration tab.



8. Select Schemas Tab & right click on mouse.

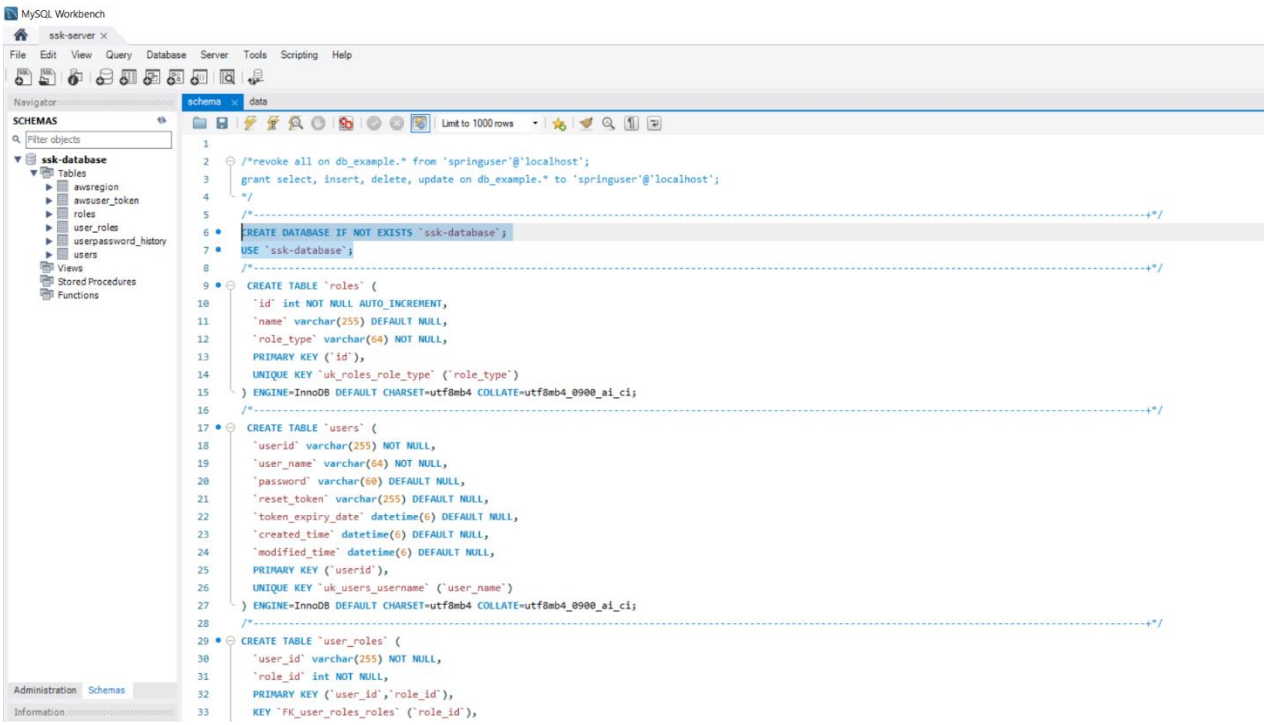


9. Click on File Tab then select “New Query Tab”



10. Edit and Execute schema.sql

- Go to File ->Open SQL Script…, navigate to the location of ‘schema.sql’ on your PC, and select it
- Modify lines 6 and 7 and choose a unique name for the schema database, for example `ssk-database`
- Make note of this name
- Highlight lines 6 and 7 as shown below and click the yellow bolt one time to execute the selected lines in schema.sql 

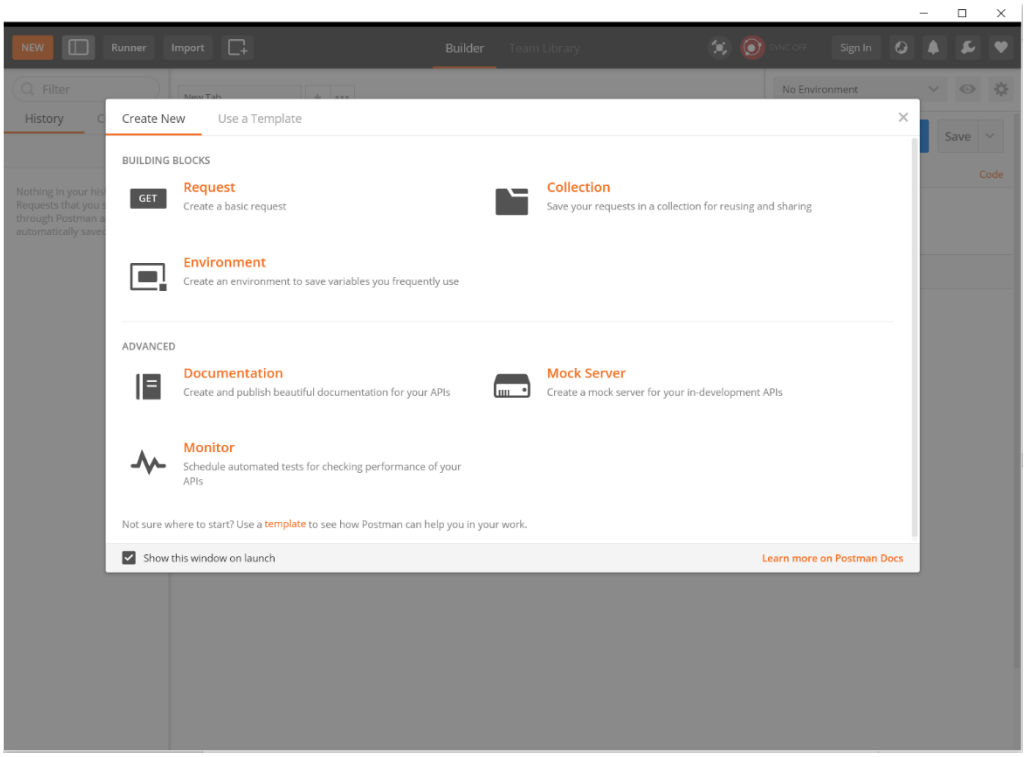


11. Edit and Execute data.sql

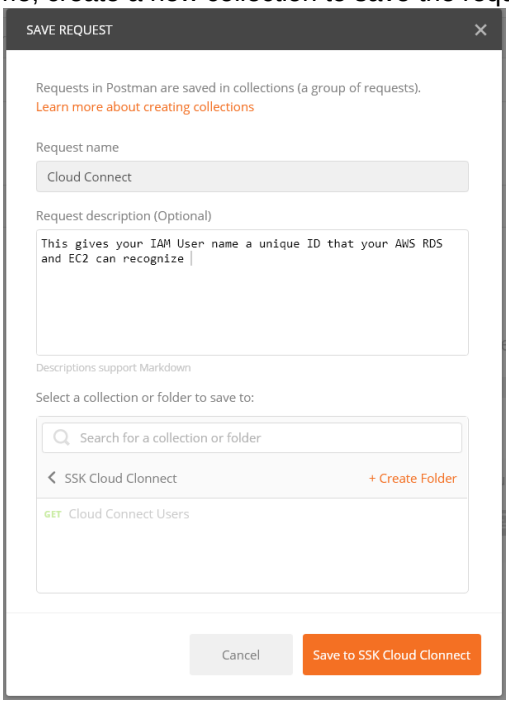
- Go to File->Open SQL Script…, navigate to the location of data.sql on your PC, and select it
- Locate the User name, Access key ID, and Secret access key of your IAM user. Note these credentials can be found in the ‘new_user_credentials.csv’ file that was downloaded after creating an IAM user
- Modify line 18 of data.sql by entering your IAM credentials. It should have a similar structure shown below

  INSERT IGNORE INTO awsuser_token(user_name,access_key_id,secret_access_key,is_root_account,create_date) values('<your_iam_username>','<your_access_key>','<your_secret_key>',1,now());

- Launch the Postman app
- Under the ‘Create New’ tab select ‘Request’



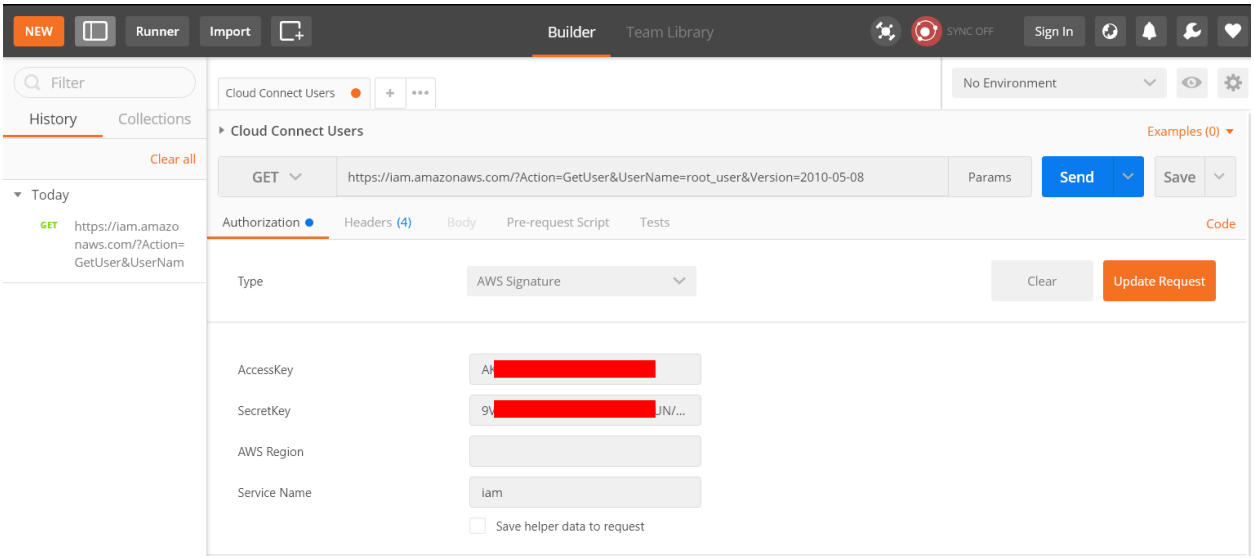
- Enter a request name, create a new collection to save the request to, and save it



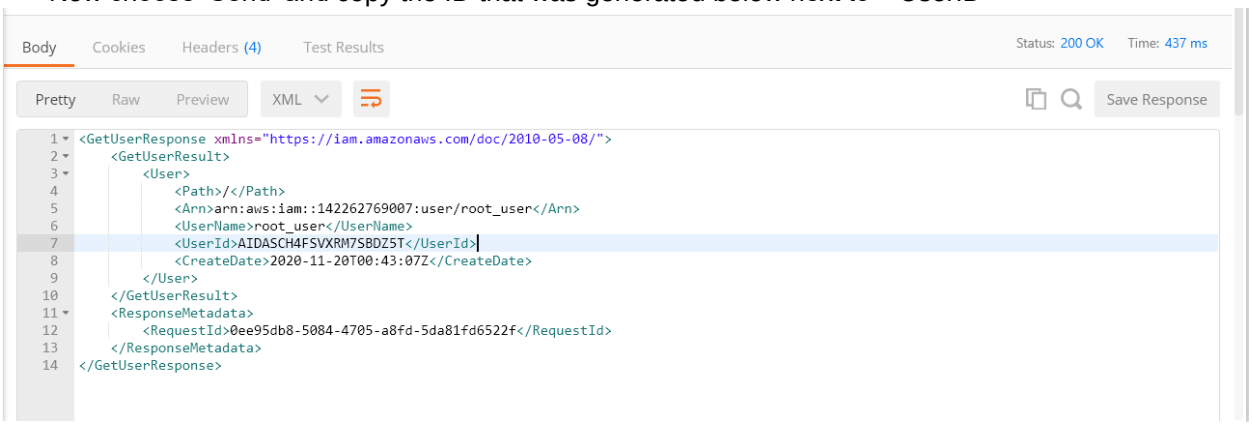
- Under the ‘Authorization’ tab select ‘AWS Signature’ next to Type and enter your IAM credentials into the ‘AccessKey’ and ‘SecretKey’ text boxes
- Next to ‘Get’ <Enter request URL> copy and paste the below URL and change the highlighted text with the username of your IAM

https://iam.amazonaws.com/?Action=GetUser&UserName=IAM-username&Version=2010-05-08

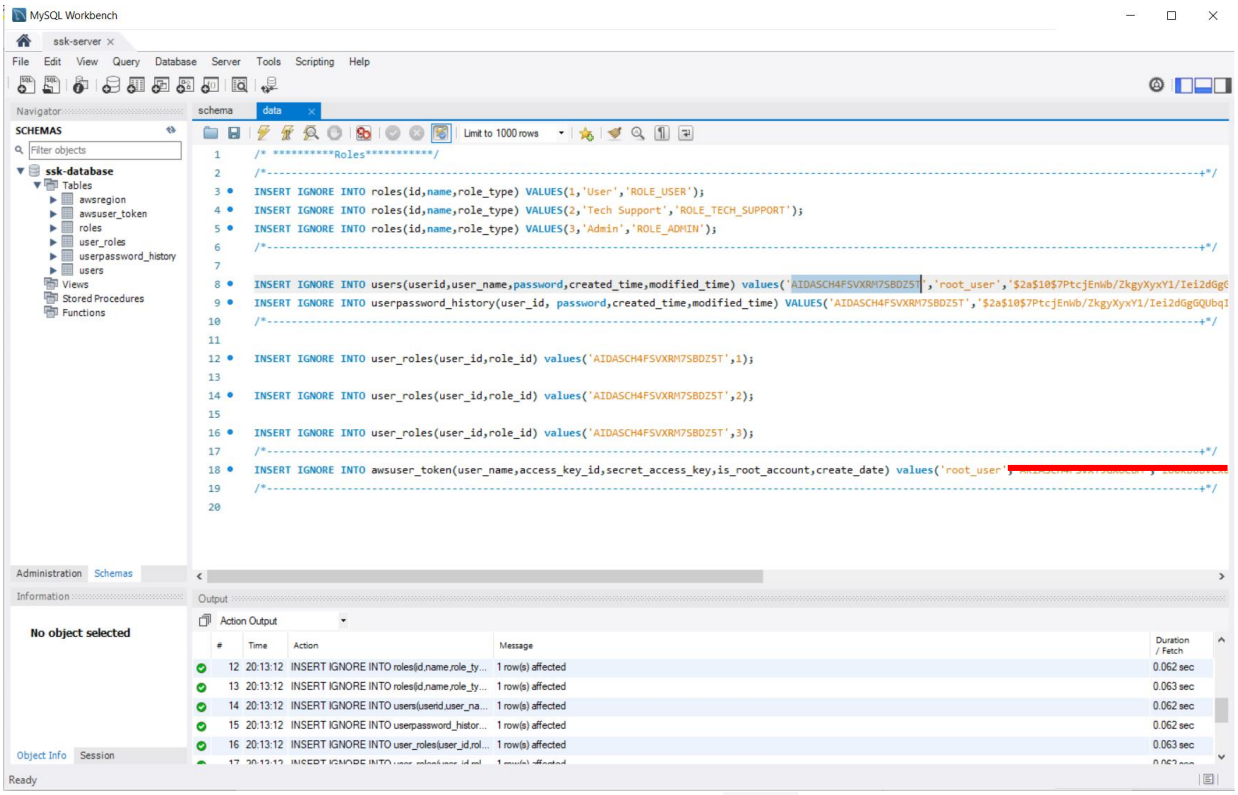
- Leave 'AWS Region' empty and enter 'iam' (all lowercase) next to Service Name as shown below



- Now choose 'Send' and copy the ID that was generated below next to '<UserID>'



- In MySQL Workbench modify data.sql by replacing the generic User ID in lines 8, 9, 12, 14, and 16 with the User ID that was created above
- In line 8 also replace 'seed.dev@einfochips.com' with the username of your IAM user



- Execute ‘data.sql’ one time by clicking the yellow bolt 

5 CONFIGURE IMAGE ON DOCKER AND EC2

5.1 Execute below commands in PuTTY

Step 1: Check “seed-server” container exists and running

$sudo docker ps -a

Step 2: Stop server & delete container (Only execute if exists & running otherwise skip it)

$sudo docker stop ssk-server
$sudo docker rm ssk-server

Step 3: Check “seed-server” image exists

$sudo docker images

Step 4: Delete image (Only execute if exists otherwise skip it)

$sudo docker rmi arrowelectronics/ssk

Step 5: Pull latest image

$sudo docker login -u <username> -p <password>
$sudo docker pull arrowelectronics/ssk:latest

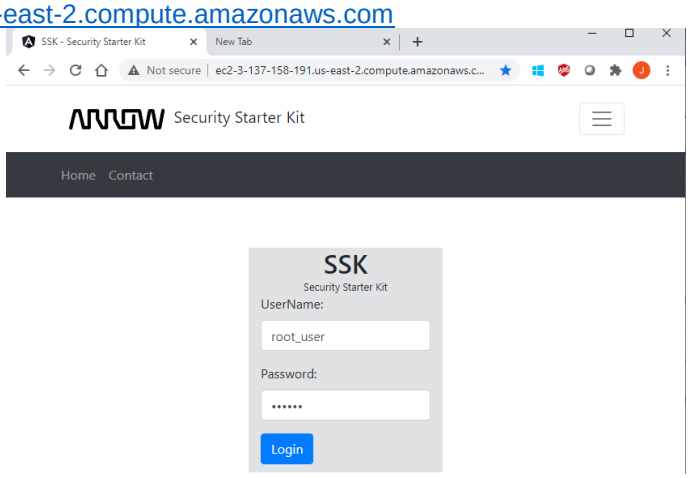
Step 6: Run image (Change highlighted text with your own values)

$sudo docker run –name ssk-server -e MYSQL_URL=”jdbc:mysql://<RDS_ENDPOINT>:3306/<SCHEMA_NAME>?useSSL=false&serverTimezone=UTC&useLegacyDatetimeCode=false&allowPublicKeyRetrieval=true” -e MYSQL_UNAME=”<RDS_UserName>” -e MYSQL_PASSWORD=”<RDS_Password>” -d -p 80:8080 -v /home/ubuntu/seedserver:/var/lib/ arrowelectronics/ssk

where <SCHEMA_NAME> is the name configured in schema.sql and <RDS_ENDPOINT>, <RDS_UserName>, and <RDS_Password> are the same values that were used to initially connect to MySQL Workbench

5.2 Log into SSK Cloud Connect

- Open a web browser and enter the URL that was noted from section 4.1
  - Note this is the Public IPv4 DNS address of your EC2 Instance i.e. http://ec2-3-137-158-191.us-east-2.compute.amazonaws.com



- Enter the UserName which is the name of the IAM username configured in ‘data.sql’ i.e. ‘root_user’
- Enter the default password: ArrowSSKportal@2020
- You can now login to your freshly installed SSK Cloud Connect Dashboard!
- Please refer to the SSK_Cloud_Connect Users Guide.pdf for configuring the different AWS services within the Arrow SSK Cloud Connect web-based tool.

6 CHECKOUT PROJECT

1. Execute following command
   git clone ssh://<name>@git.einfochips.com:29418/secure-end-to-end-device
   git checkout seed-server
2. Manage permission regarding project
   URL: https://git.einfochips.com:8080/q/status:open

7 BUILD PROJECT

7.1 Execute below command

1. Build Angular project (Go to /secure-end-to-end-device/seed-client)
   ng build --prod
2. Copy build file under static folder
   From path /secure-end-to-end-device/seed-client/dist/seed-client/
   To
   /secure-end-to-end-device/seed-cloud/seed-server/src/main/resources/static
3. To Build Spring boot application execute following command
   ./gradlew clean build
4. To Build image (Go to /secure-end-to-end-device/seed-cloud/seed-server)
   docker build -t arrowelectronics/ssk:latest .
5. Push image to Docker hub
   docker push arrowelectronics/ssk:latest

8 REFERENCES

[1] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html

[2] https://docs.docker.com/engine/install/ubuntu/

[3] https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-18-04

[4] https://aws.amazon.com/console/