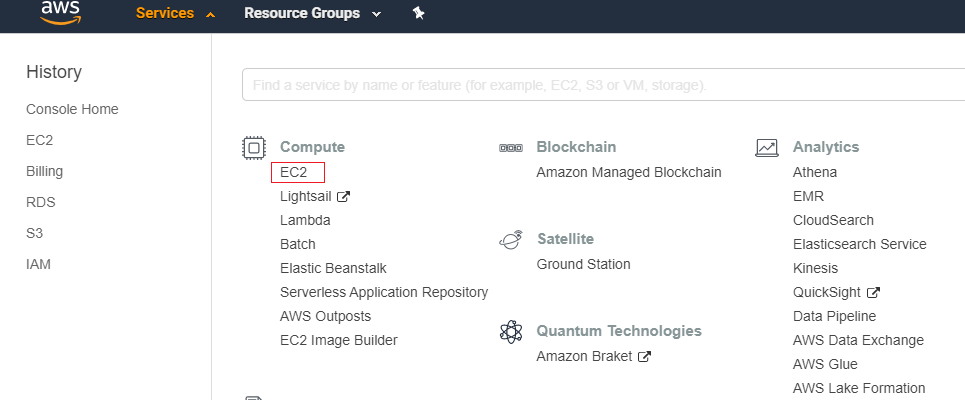
# **Food Delivery Application Deployment on AWS**

# **1. Deploying Spring Boot Application on AWS**

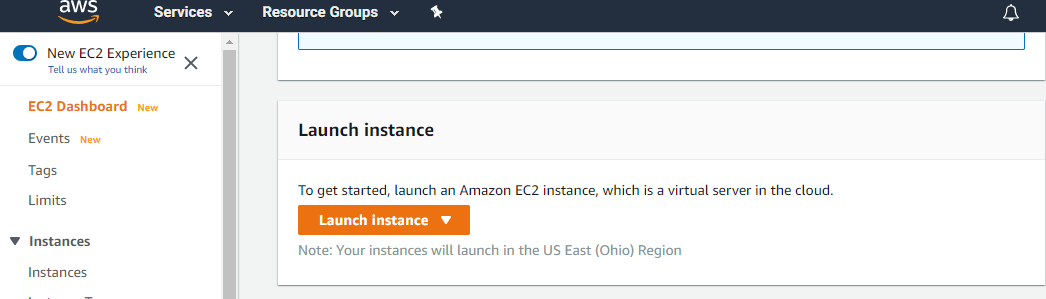
## 1.1 Launch an AWS EC2 Instance

**Step 1)** Sign in to the preview version of the AWS Management Console.

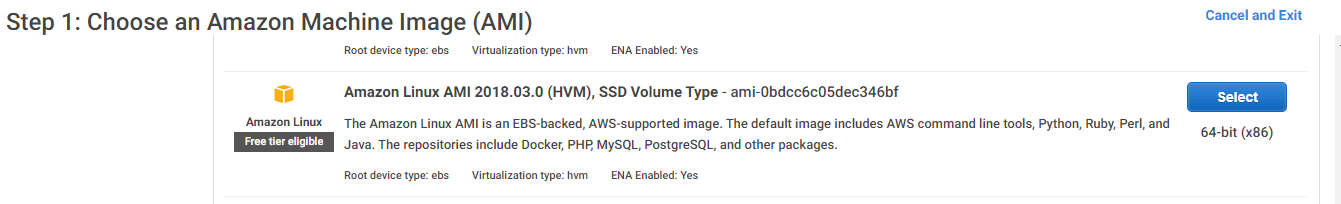
**Step 2)** Open the Amazon EC2 console by choosing EC2 under compute. If you are using Show All Services view, your screen looks like this:



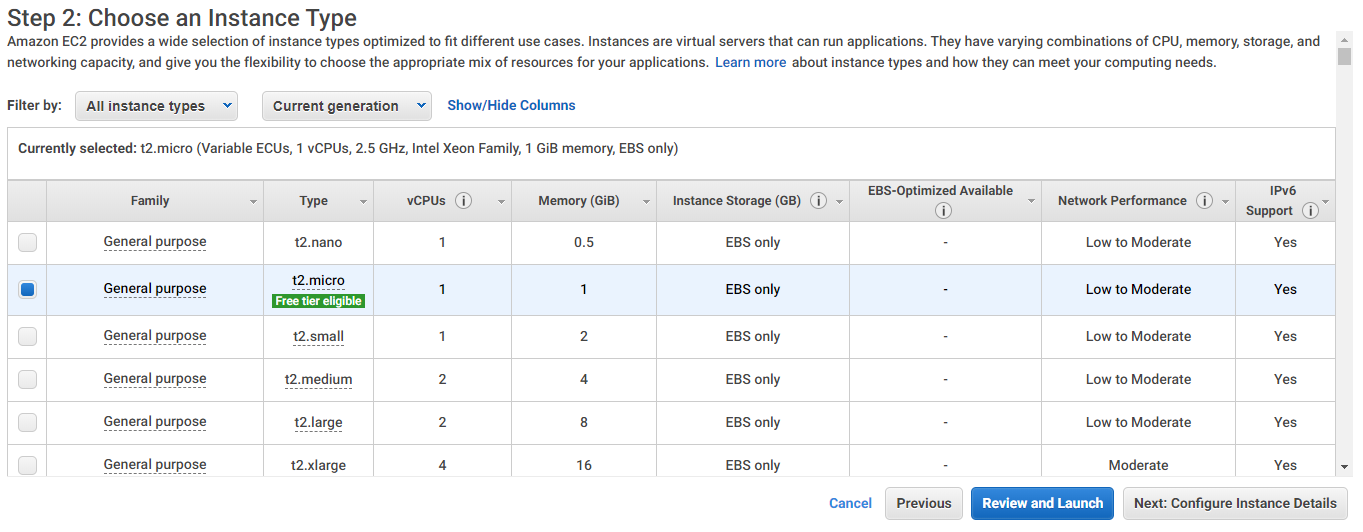
**Step 3)** From the Amazon EC2 dashboard, Choose Launch Instance.



**Step 4)** The choose an Amazon Machine Image (AMI) page displays a list of basic configurations called Amazon Machine Images (AMIs) that serve as templates for your instance. Select the Amazon Linux AMI. Notice that this configuration is marked Free tier eligible.

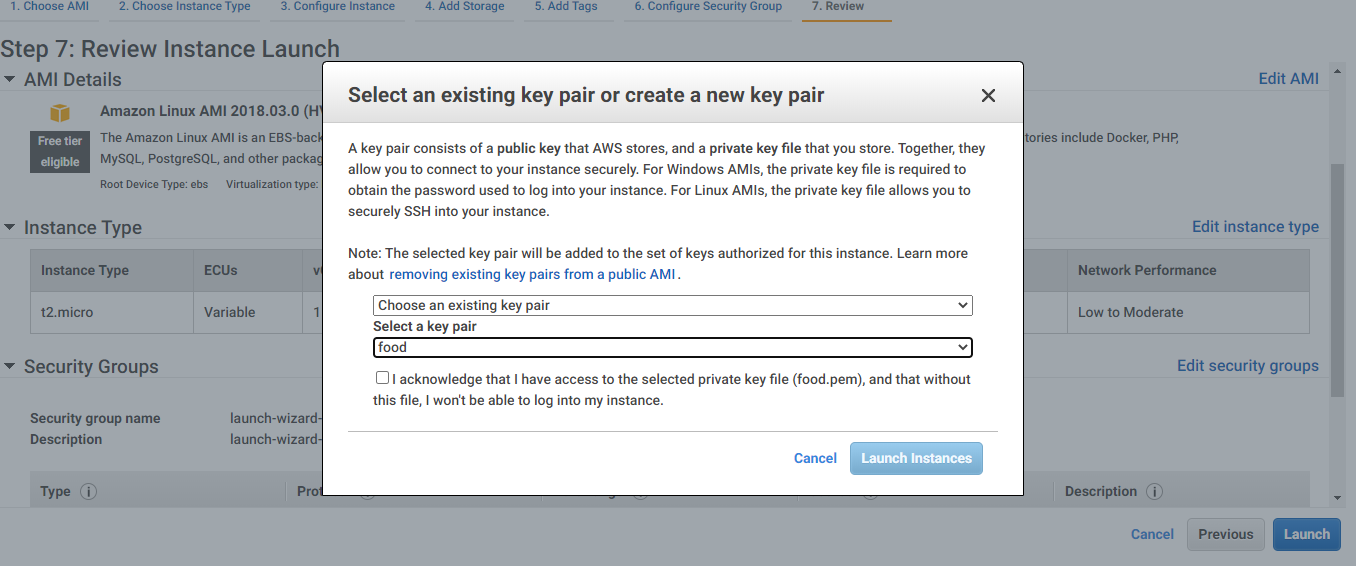


**Step 5)** On the Choose an Instance Type page, choose t.2micro as the hardware configuration of your instance and Review and Launch.

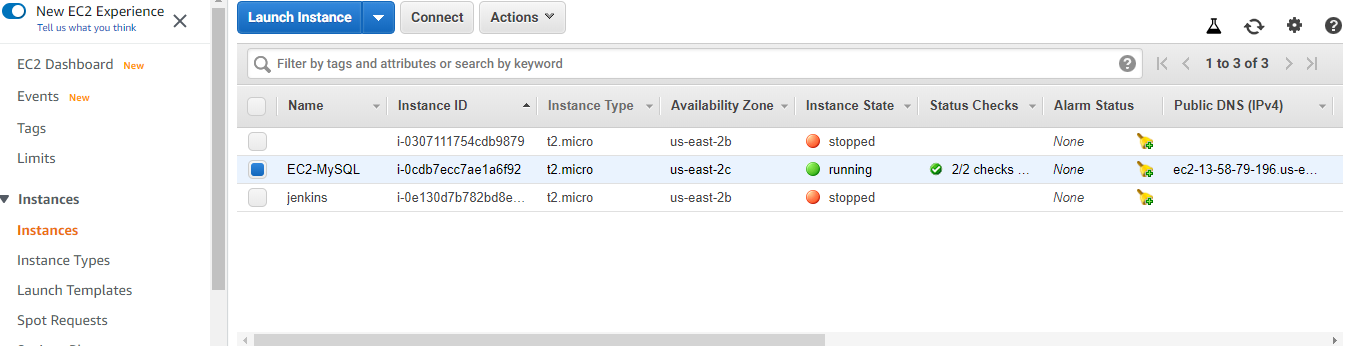


**Step 6)** On the Review Instance Launch page, choose Launch.

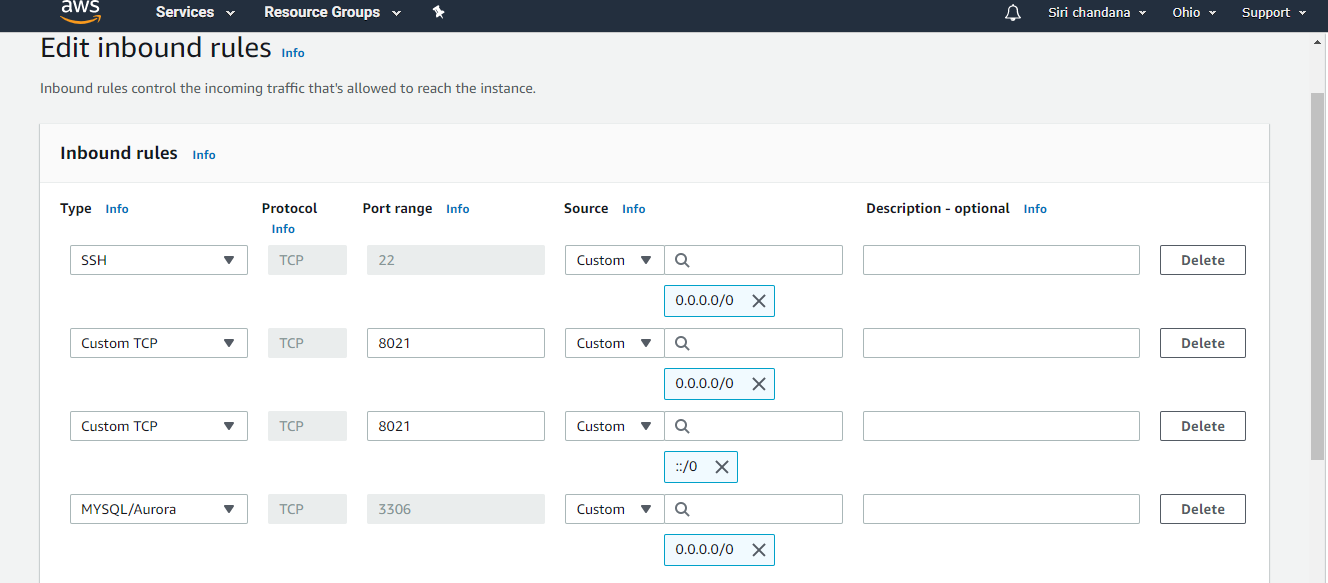
**Step 7)** In the Select an existing key pair or create a new key pair dialog box, choose Create a new key pair, enter a name for the key pair and then choose Download Key Pair.



**Step 8)** Once the instance is ready and running, it will appear as below



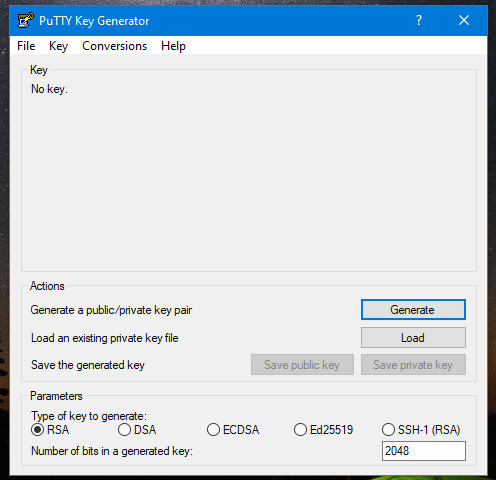
**Step 9)** Go to Security group and add the rules for MySQL and Tomcat port (port number of your application) as below



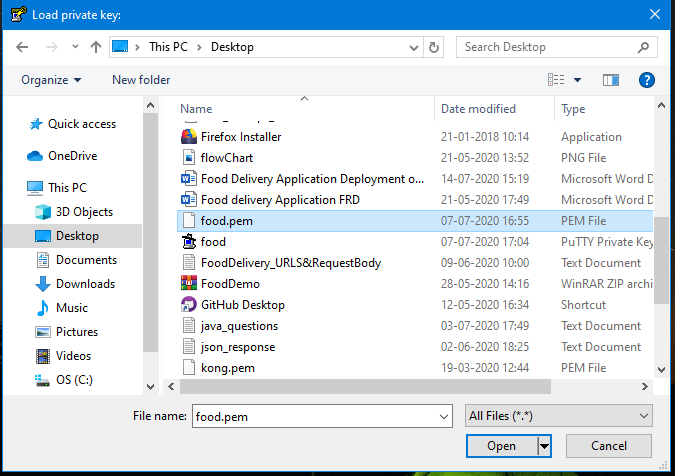
## 1.2 Connecting to Instance using PuTTY

**Step 1)** Go to <https://www.putty.org/> and download the PuTTY or PuTTYgen.

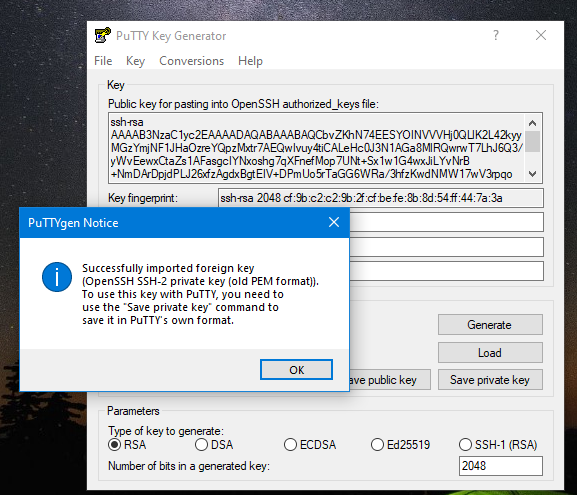
**Step 2)** Click on Start menu -> All Programs -> PuTTY -> PuTTYgen.



**Step 3)** Next, click on the option ‘Load’. As PuTTY supports its native file format, it will only show files that have .ppk file extension. Therefore, users have to choose the ‘All Files’ option from the drop-down bar. It will display all key files included the .pem file.

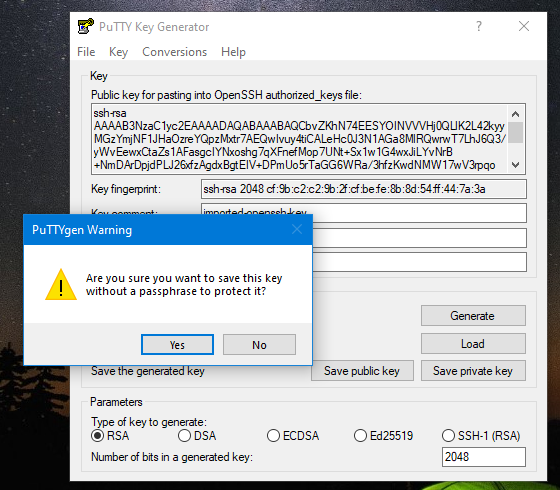


**Step 4)** Now select the .pem file that you want to convert. As PuTTYgen is used for SSH connectivity, so it crucial for users to select the specific file that they plan to convert and click ‘Open’. To confirm, click on ‘OK”.



**Step 5)** In the resultant window,click on ‘Save private key’ which will convert and save the key file in PuTTY compatible format.

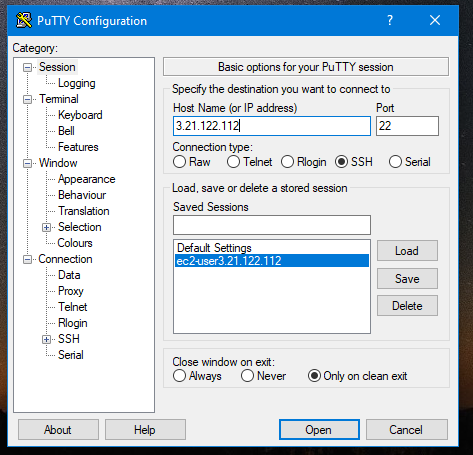
**Step 6)** PuTTYgen will prompt a warning of saving the key without a passphrase. Hit ‘Yes’ on it.



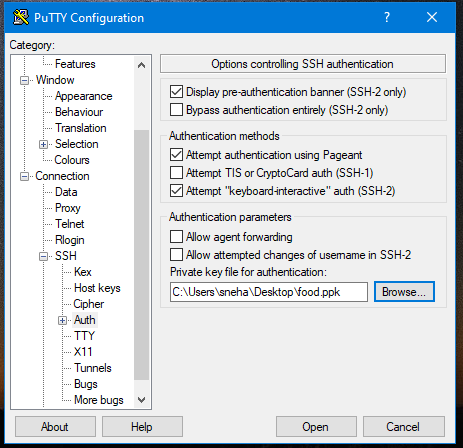
**Step 7)** Now, give the name to your file and PuTTYgen will automatically add .ppk file extension.

**Step 8)** Once the .pem file is converted .ppk then users can connect to remote hosts using PuTTY’s SSH client.

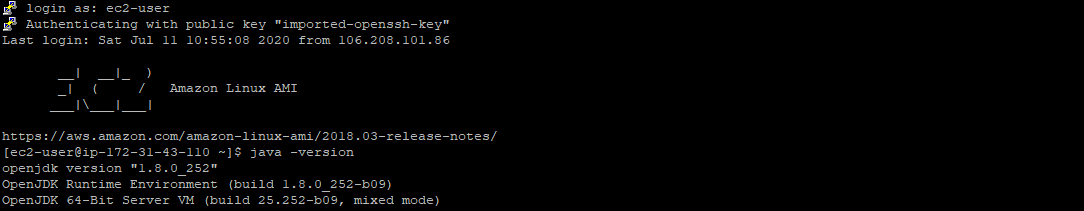
**Step 9)** First, open PuTTY and input the host IP address i.e. Public IP address of EC2 instance.



**Step 10)** Go to SSH -> Auth -> browse the ppk file, select the file and click open.

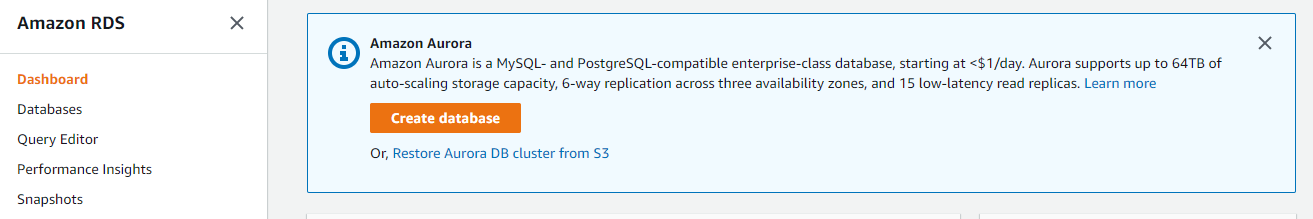


**Step 11)** Once the PuTTY is connected, login as ec2-user

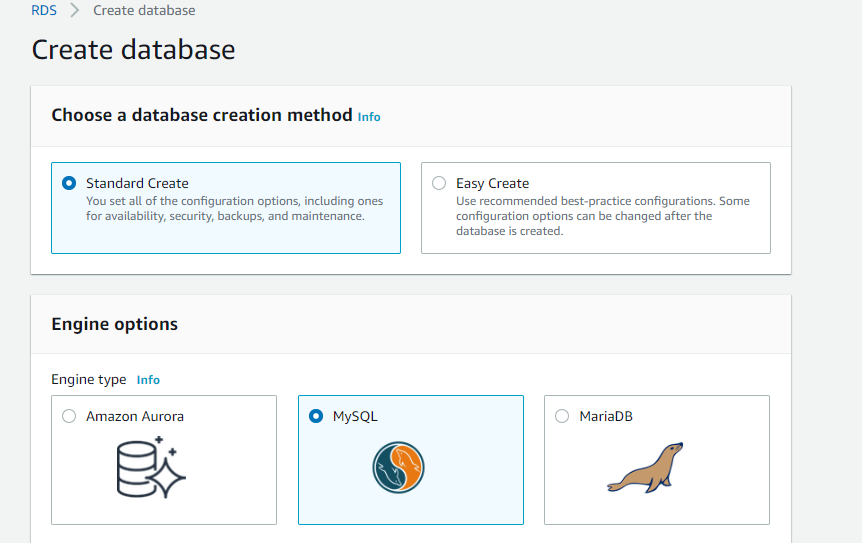


## 1.3 RDS Database

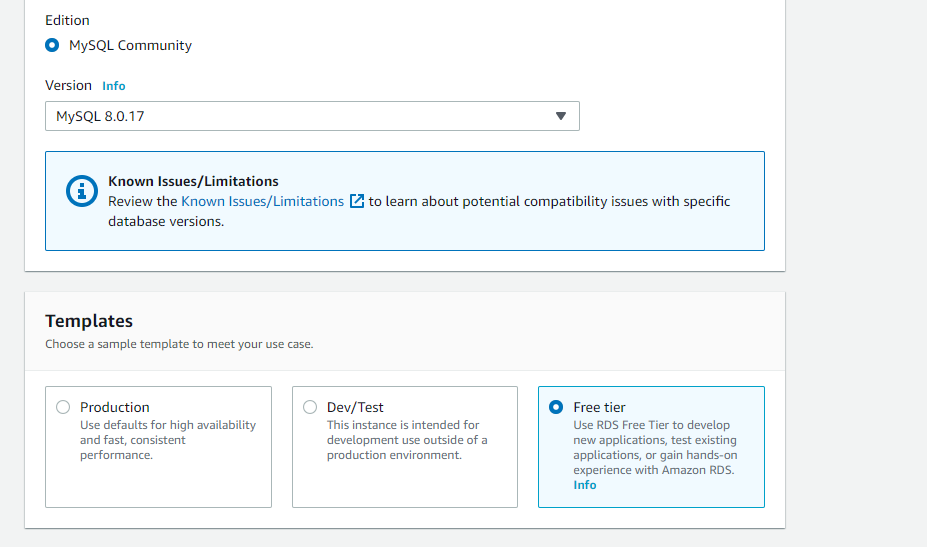
**Step 1)** Go to Amazon RDS and choose Create database.



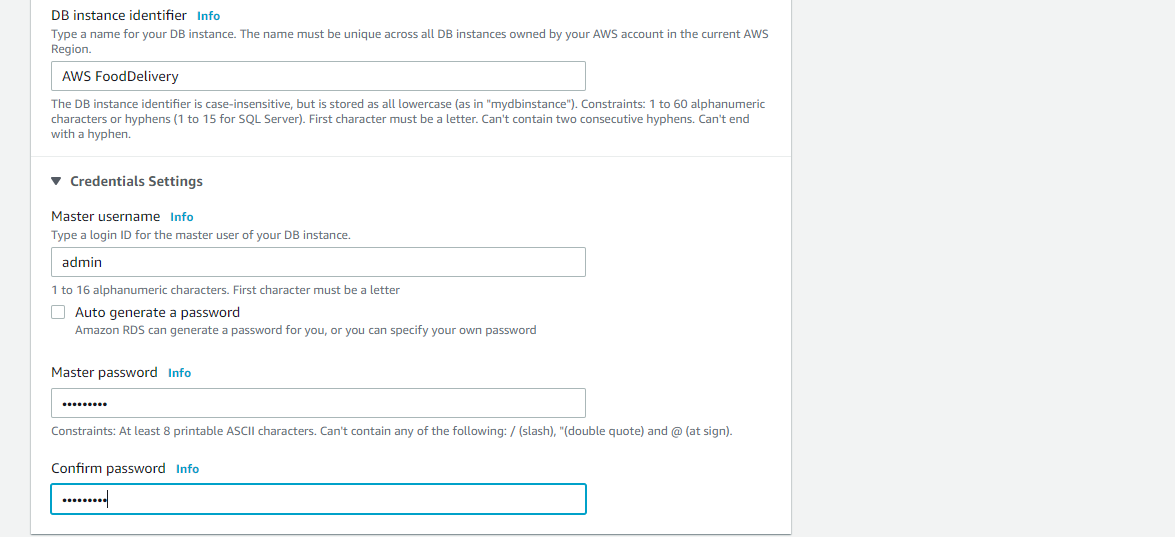
**Step 2)** Click the MySQL icon.



**Step 3)** Leave the default value of edition and engine version and select the Free Tier template.

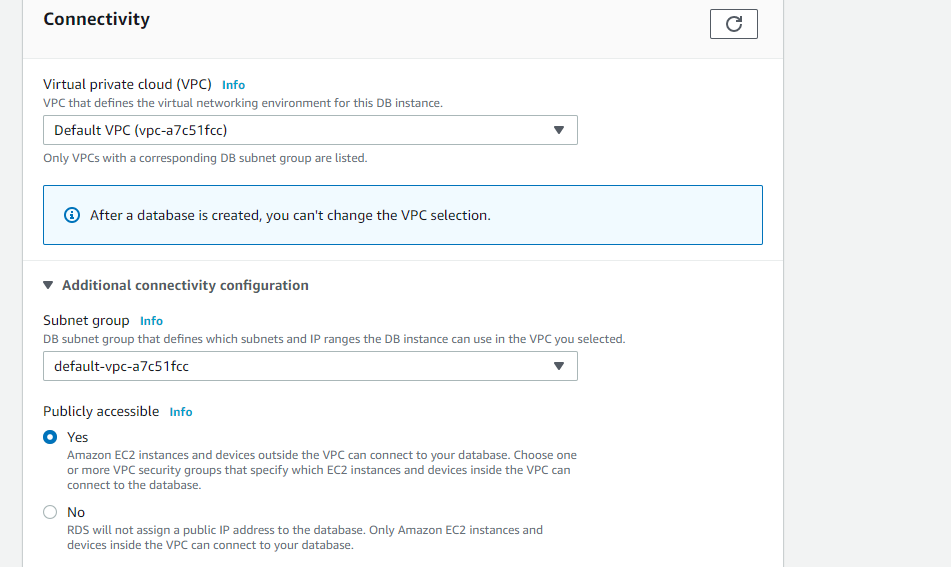


**Step 4)** Configure your DB instance by setting username and password

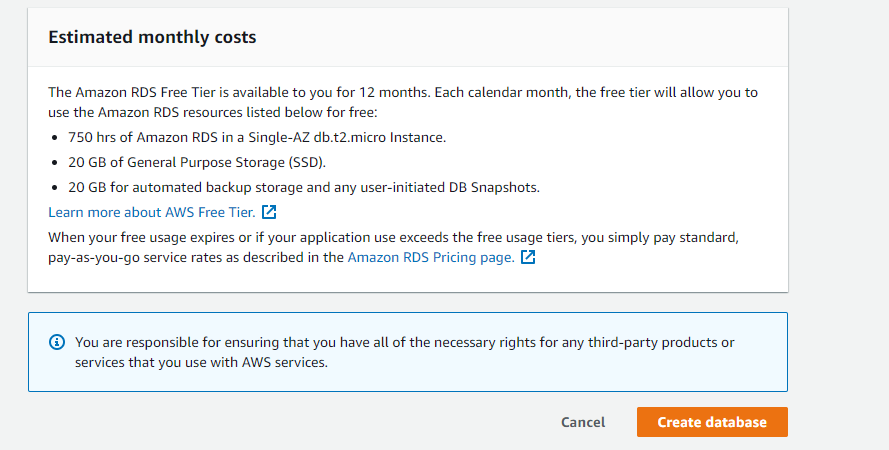


**Step 5)** In Connectivity section, make Public accessibility to Yes.

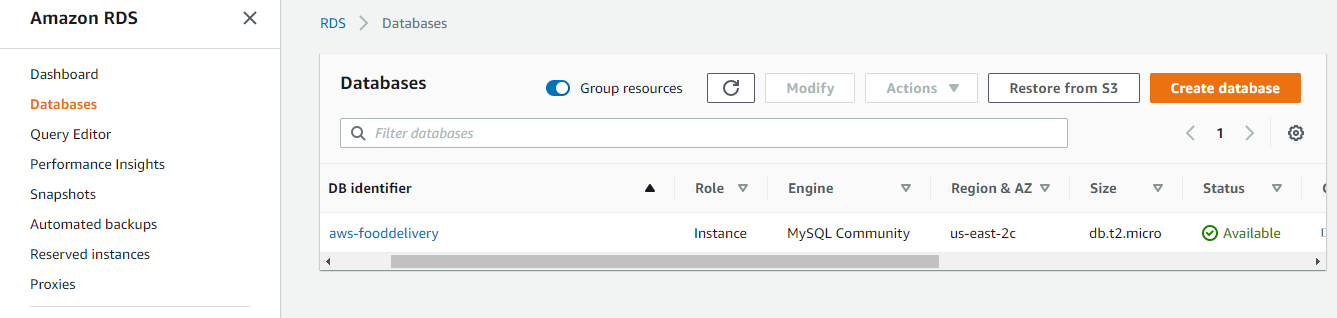
This will allocate an IP address for your database instance so that you can directly connect to the database from your own device.



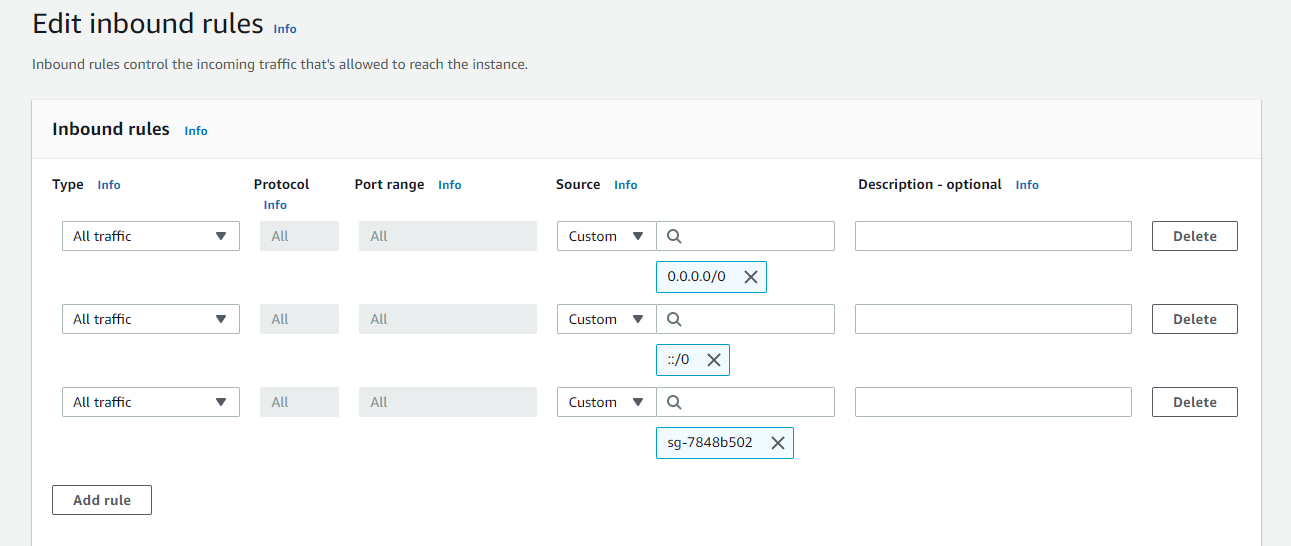
**Step 6)** Click Create Database



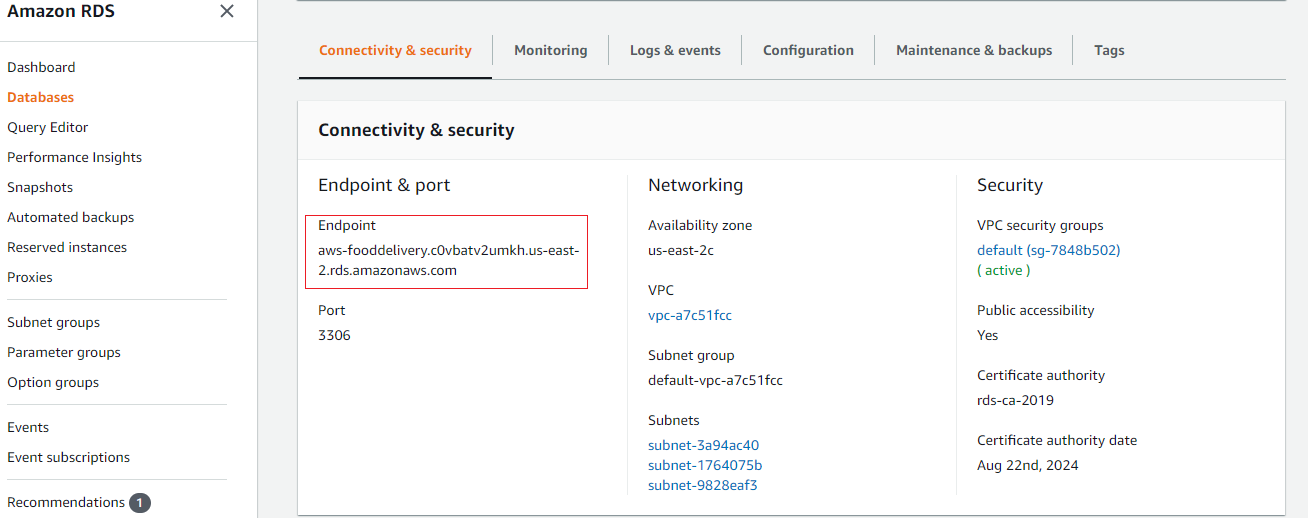
**Step 7)** Once the Database is ready, it will appear as Available



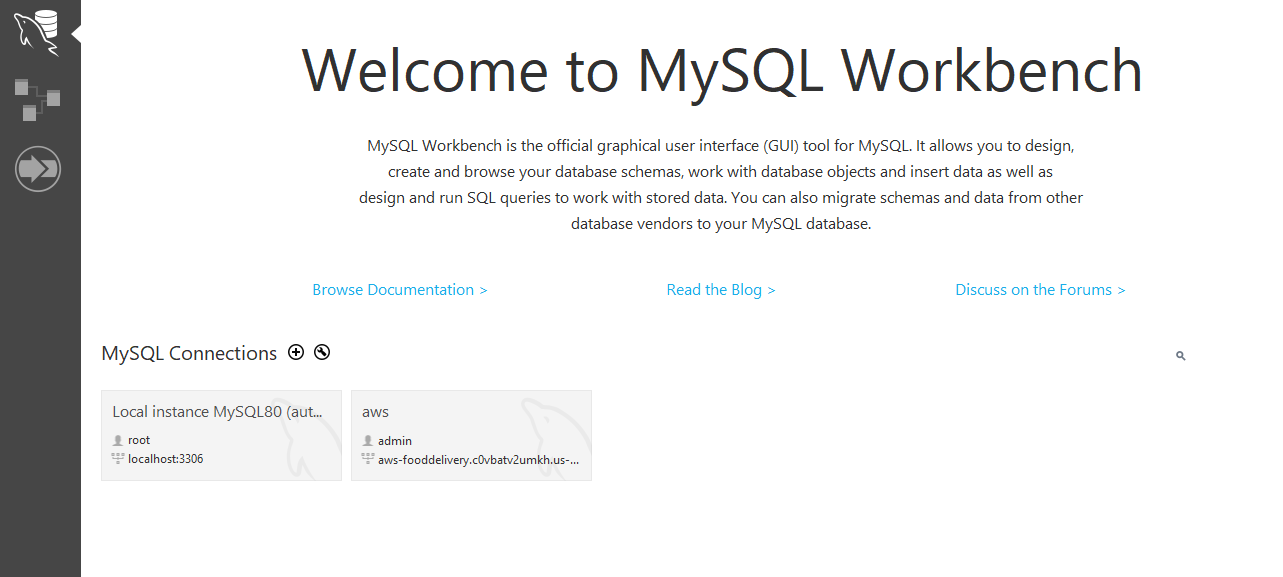
**Step 8)** Go to Security group and add rule All traffic and select the default RDS security group.



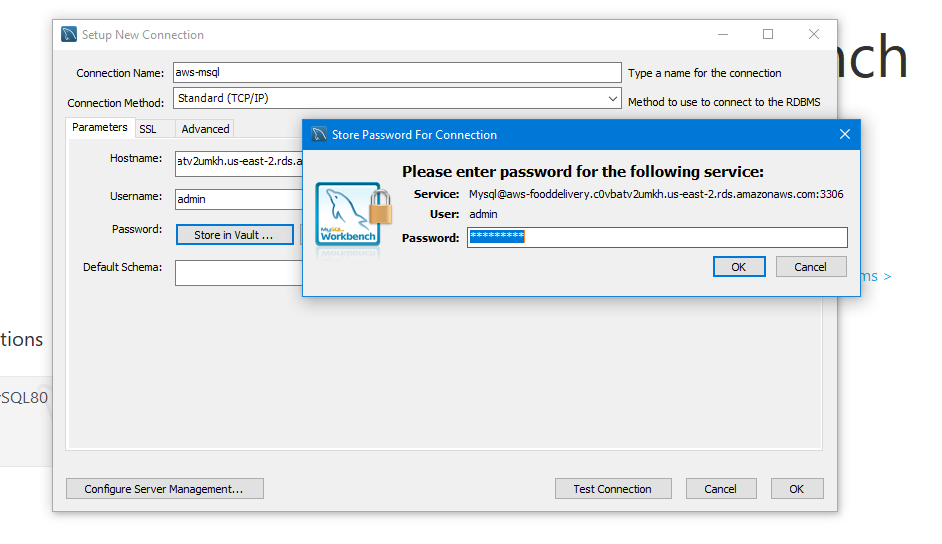
**Step 9)** Copy the Endpoint as it is used to connect the local MySQL workbench



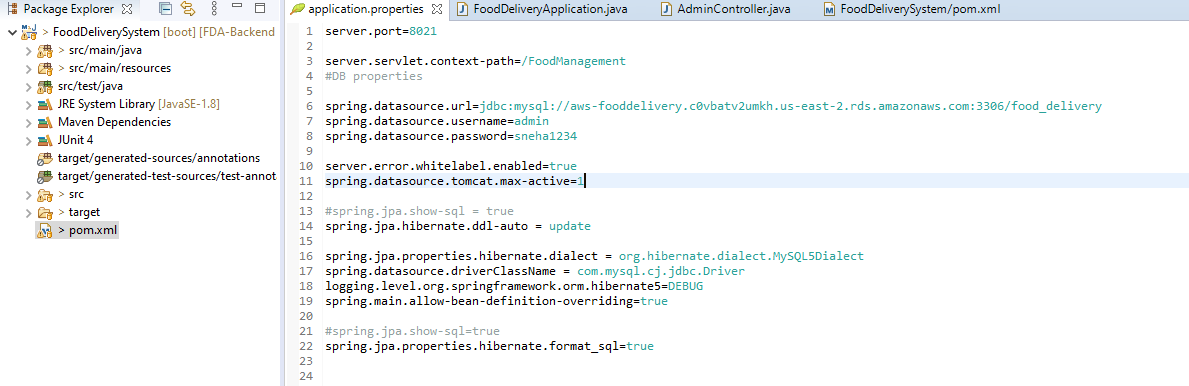
**Step 10)** Go to MySQL Workbench and in MySQL Connections add the RDS details



**Step 11)** In Hostname: give the Endpoint, provide the Username and password as per RDS database and click on Test Connection.



**Note:** In application.properties file add the RDS Mysql connections



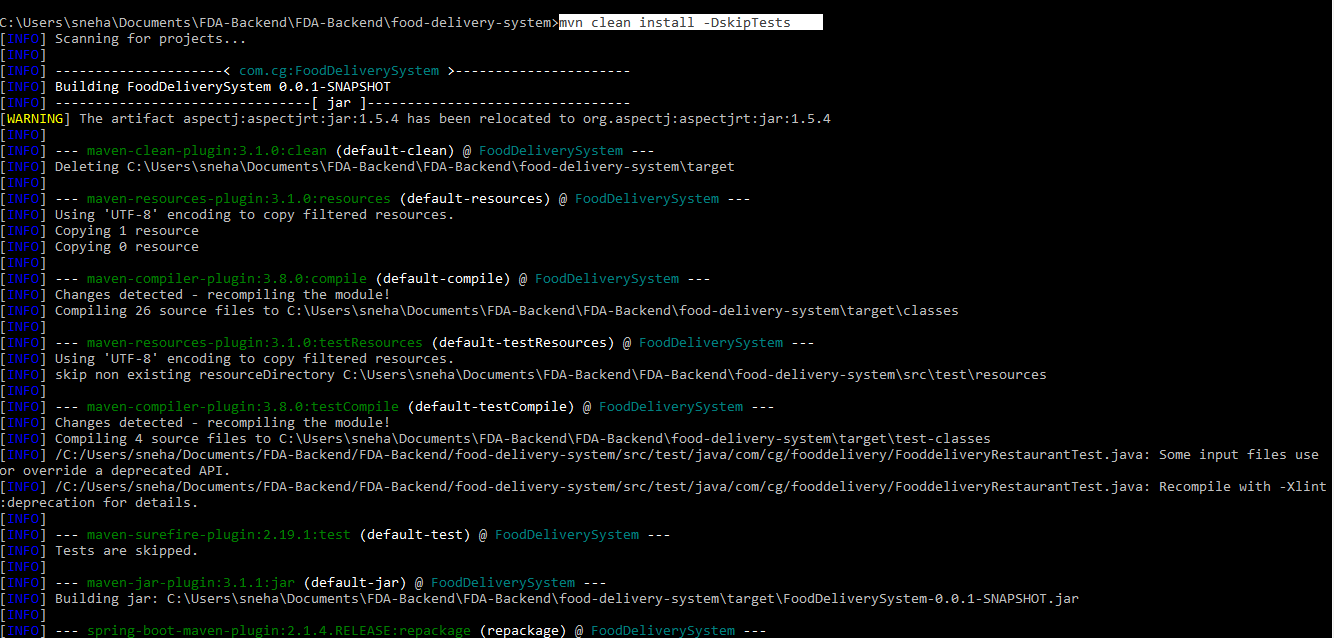
## 1.4 Generating jar file

**Step 1)** Go to the pom.xml file location in system explorer.

**Step 2)** Type cmd and open the command prompt.

**Step 3)** Give the command as mvn clean install -DskipTests

**Step 4)** Once the build is success, the jar will be generated in target folder.

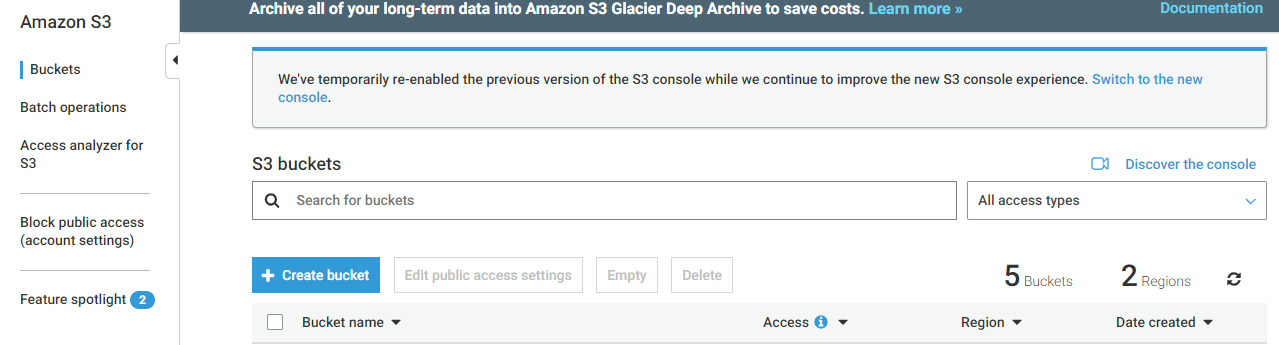


## 1.5 Create S3 Bucket in AWS and upload jar

First, you need to create an Amazon S3 bucket where you will store objects.

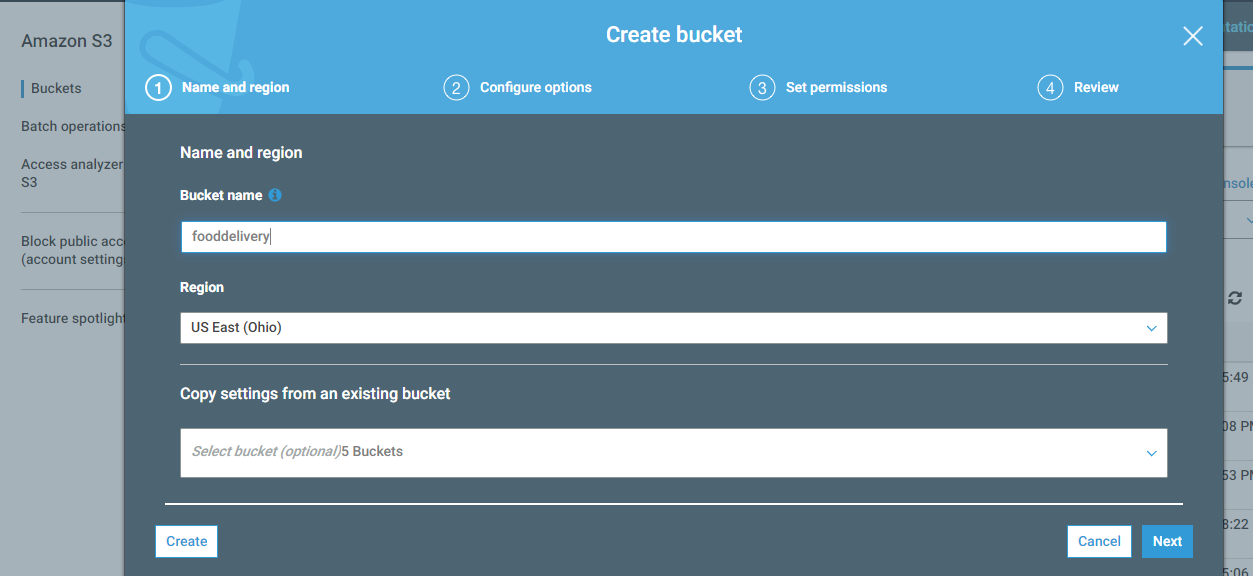
**Step 1)** Under Storage and Content Delivery, choose S3 to open the Amazon S3 console.

**Step 2)** From the Amazon S3 console dashboard, choose Create Bucket.



**Step 3)** In Create a Bucket, type a bucket name in Bucket Name.

The bucket name you choose must be globally unique across all existing bucket names in Amazon S3 (that is, across all AWS customers).

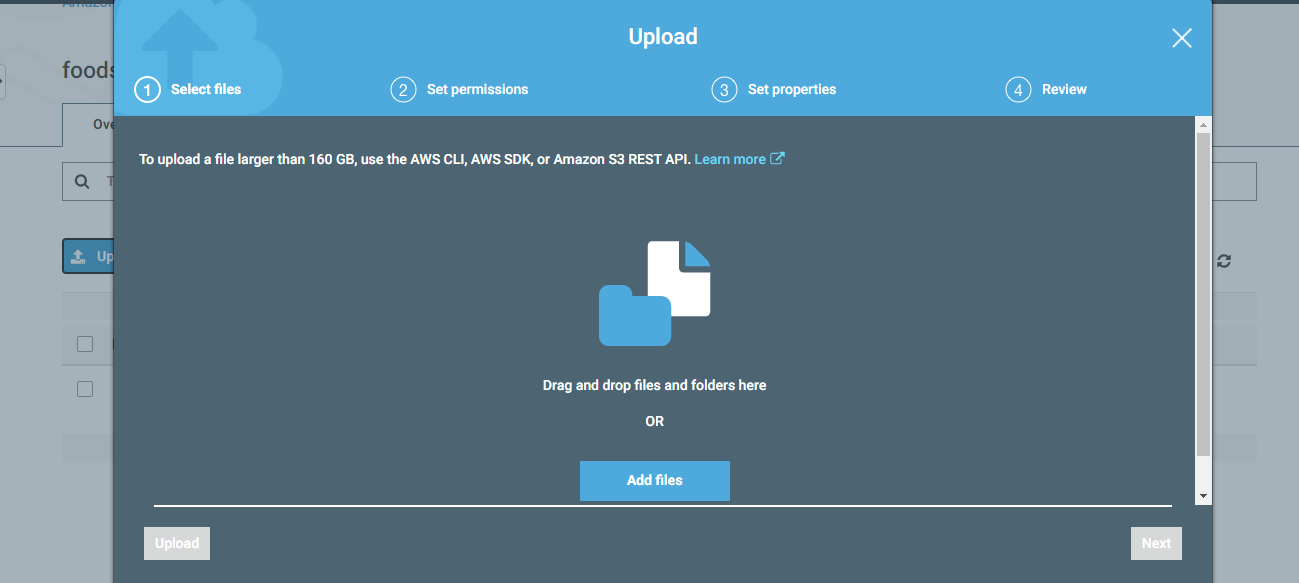


**Step 4)** Choose create

When Amazon S3 successfully creates your bucket, the console displays your empty bucket in the Buckets pane.

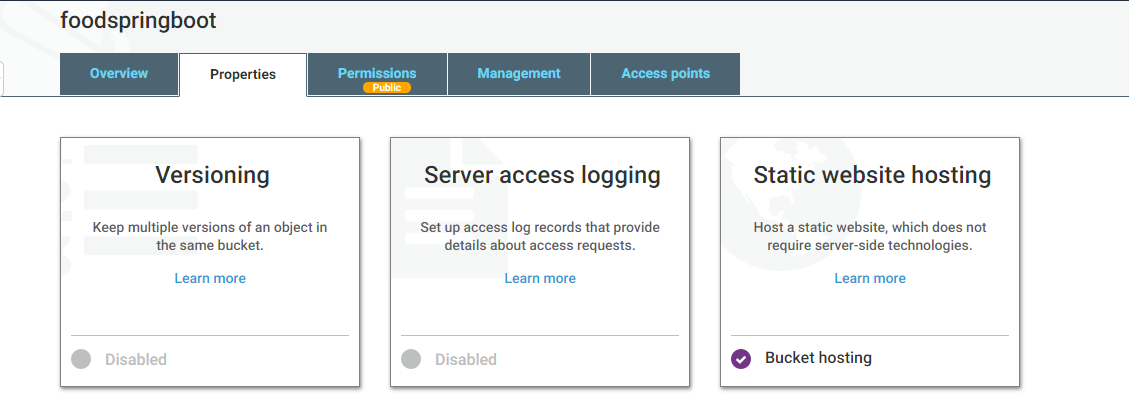
**Step 5)** Upload the jar file

You can directly Drag and drop files and folders or click on Add files to select the file.

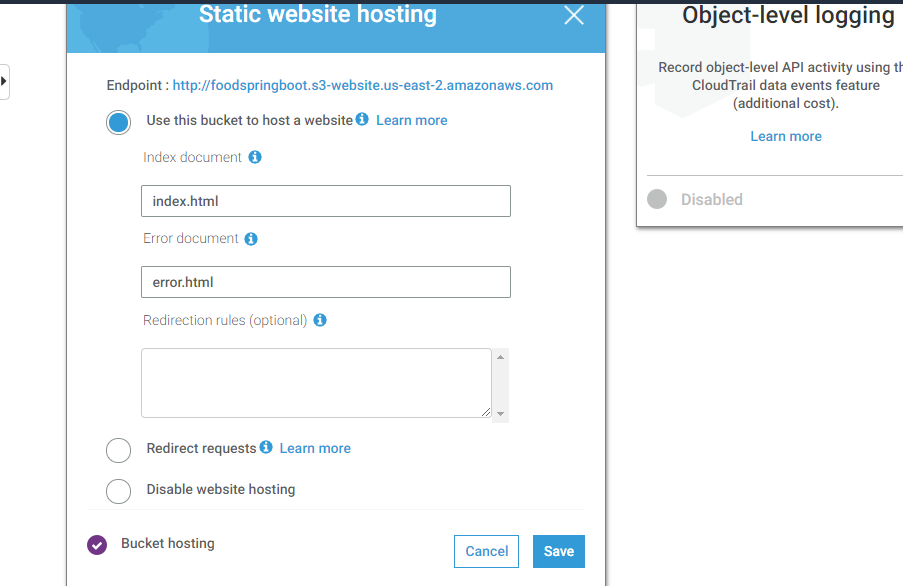


**Step 6)** Allow S3 as Public Access

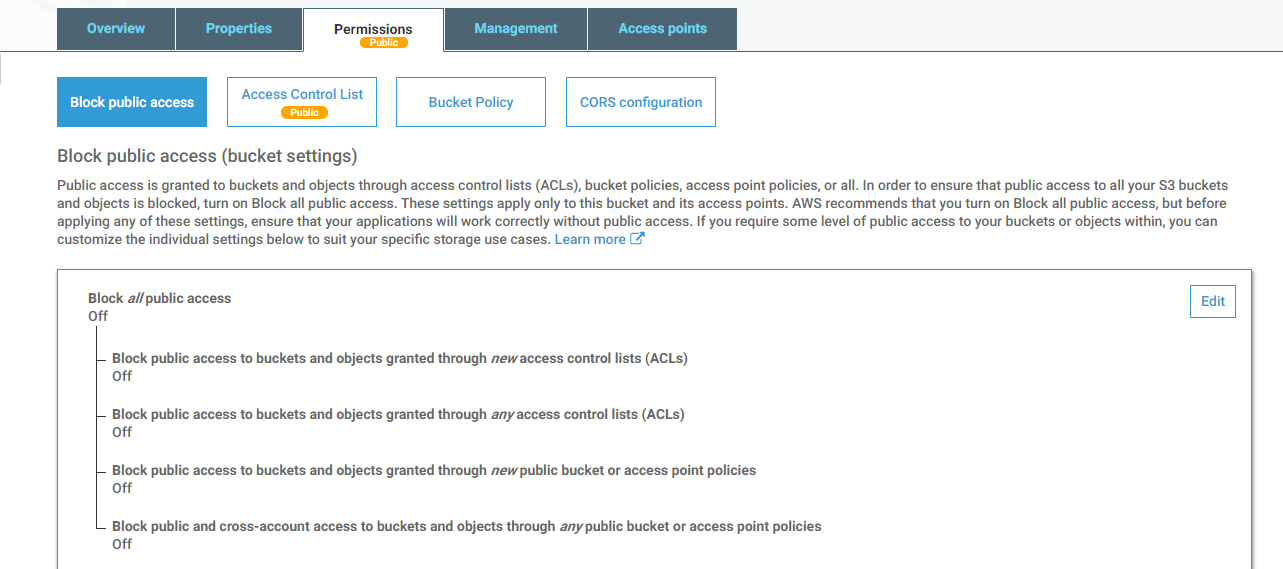
Go to Properties, select the Static website hosting



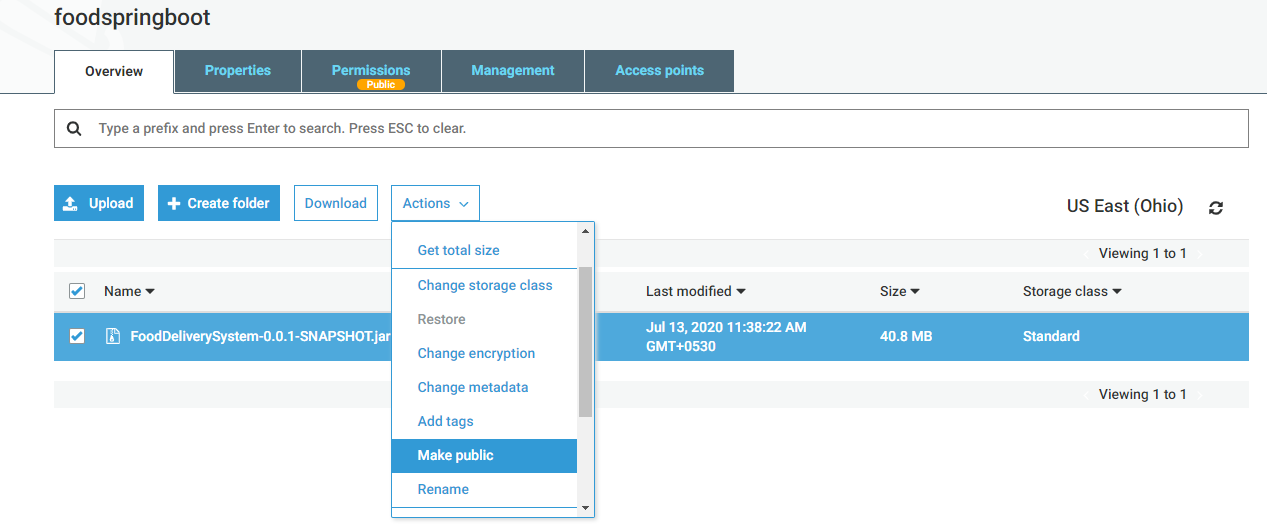
**Step 7)** In Static website hosting, make changes as below and click on save.



**Step 8)** Go to Permissions, click on Edit and turn off the Block all public access.



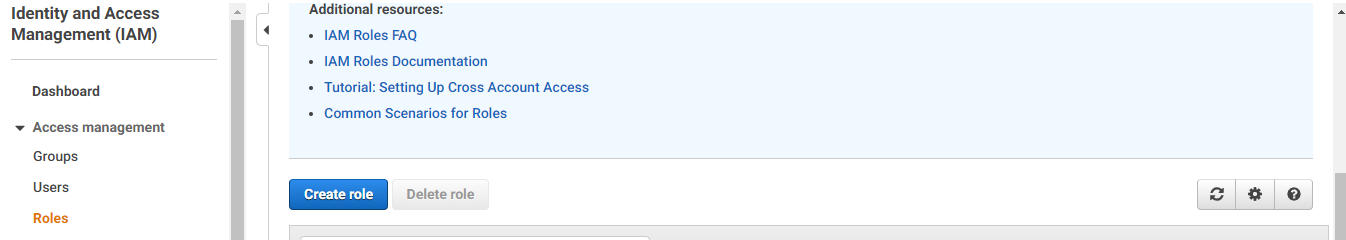
**Step 9)** After uploading jar in S3 Bucket, click on Actions and Make jar as public.



## 1.6 Create IAM Role

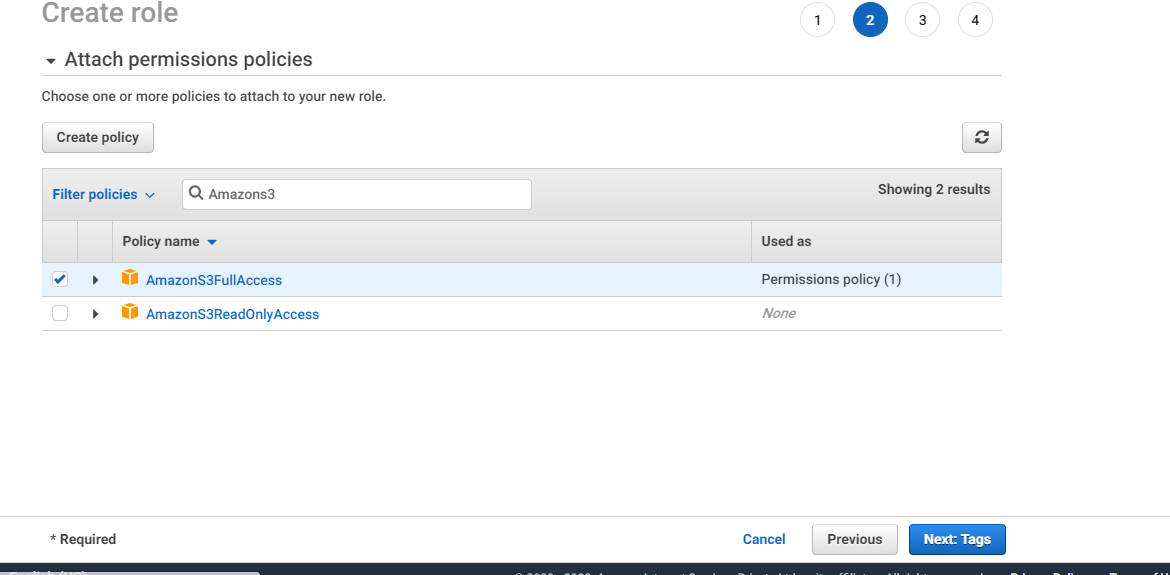
Creating a Role for service using the AWS Management Console.

**Step 1)** In the navigation pane of the console, click Roles and then click on “Create Role”.

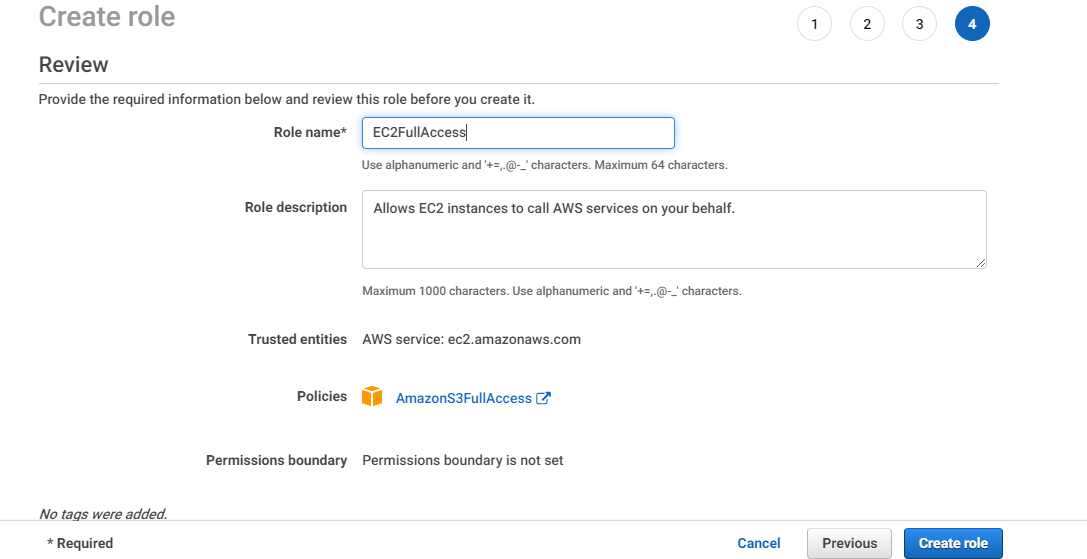


**Step 2)** Choose the service that you want to use with the role. In my case EC2.

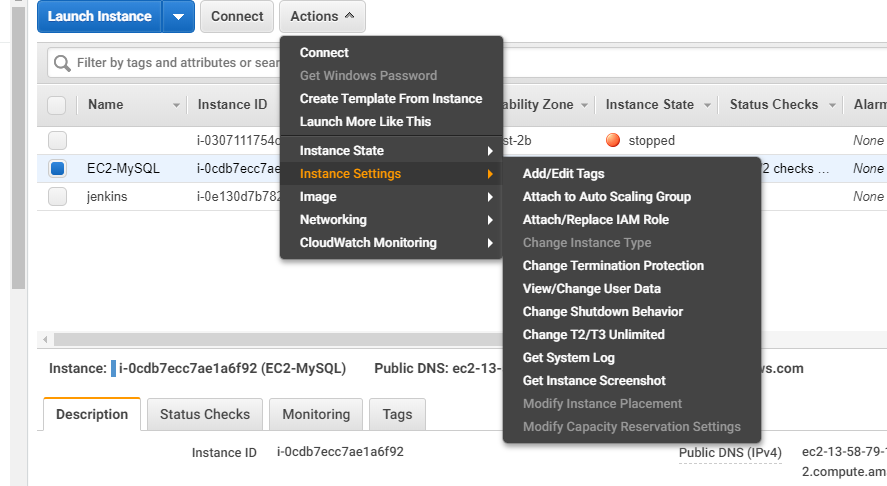
**Step 3)** Select the managed policy that attaches the permissions to the service.



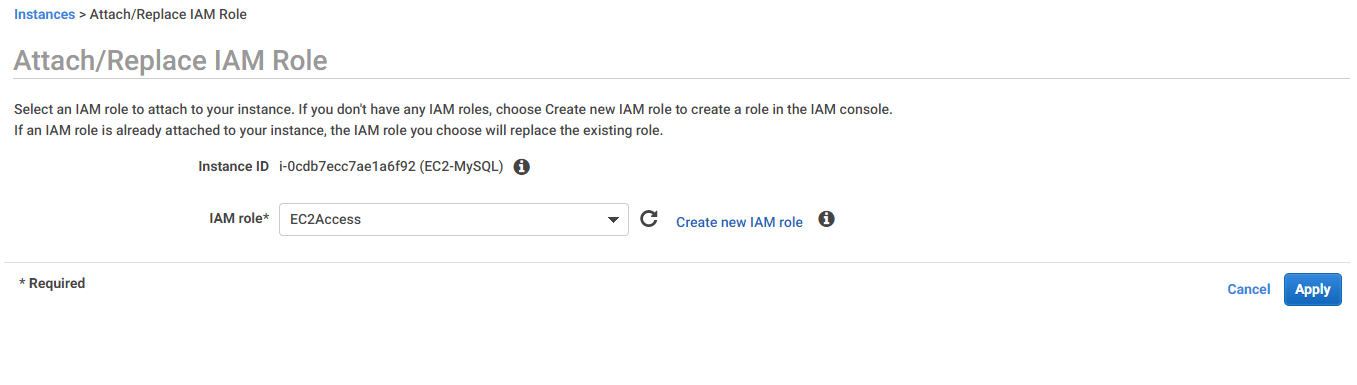
**Step 4)** In a role name box, enter the role name that describes the role of the service and then click on “Create role”.



**Step 5)** Go to EC2 instance and Attach the IAM role.



**Step 5)** Select the created IAM Role and click on Apply.



## 1.7 Copy jar from S3 to EC2 instance

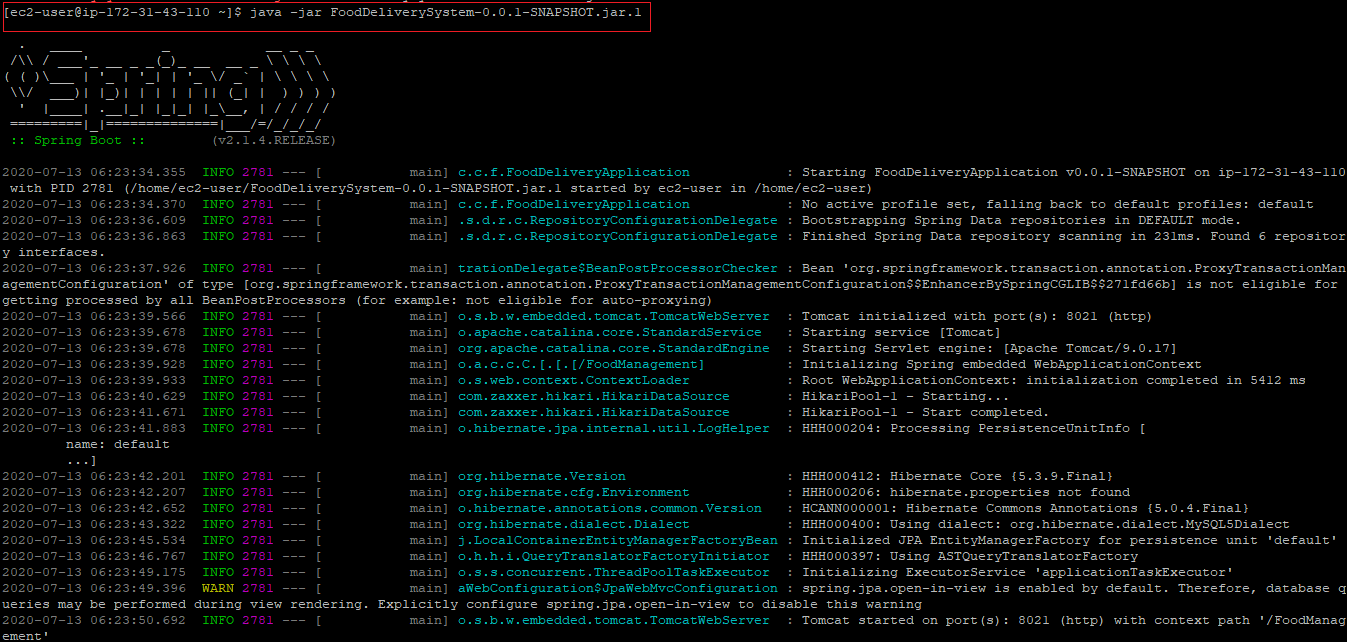
**Step 1)** Once the jar is successfully uploaded in S3, we can copy the jar into EC2 instance using below command.



**Step 2)** We can check the jar is copied or not using below command.



**Step 3)** Run the jar using below command.



**Step 4)** Once the application is started, we can test the output as below

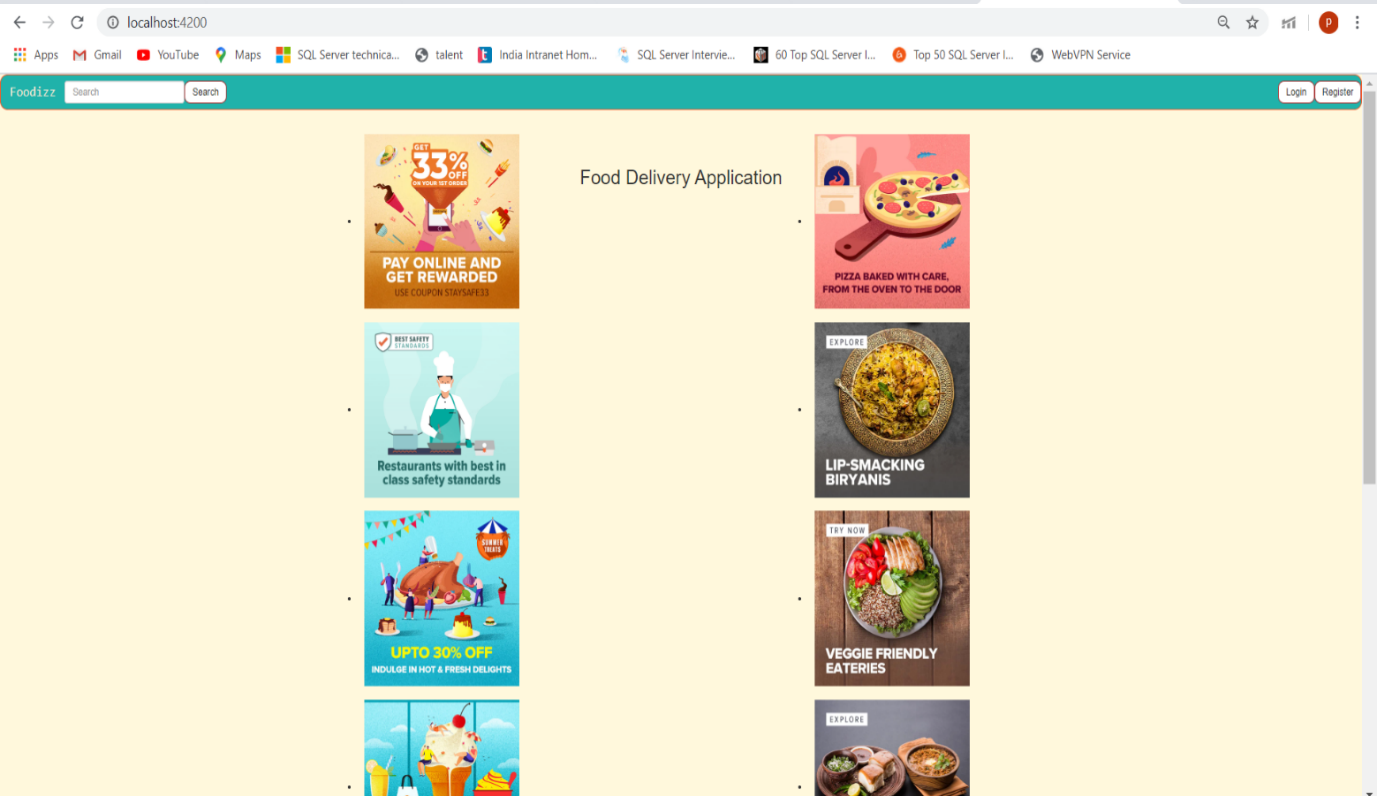
<http://ec2-13-58-79-196.us-east-2.compute.amazonaws.com:8021/FoodManagement/getAdmins>

# **2. Deploying Angular Application on AWS**

* run ng build in Visual studio code terminal
* dist folder will be made in the project folder
* Go to AWS S3 bucket
* upload the dist file there
* S3 Bucket will give you the url

<http://fooddelivery5.s3-website-us-east-1.amazonaws.com/>

* run it in chrome clearing all the cache



# **3. Team Members**

* Palakurthy Spandana
* Sneha Manga
* Kongara BinduSri
* Kamble Aditya
* Pooja Tiwari