AWS Bean Stalk Deploy MONO PERN Stack

1. Login to AWS Console create a free account if you don't have one already

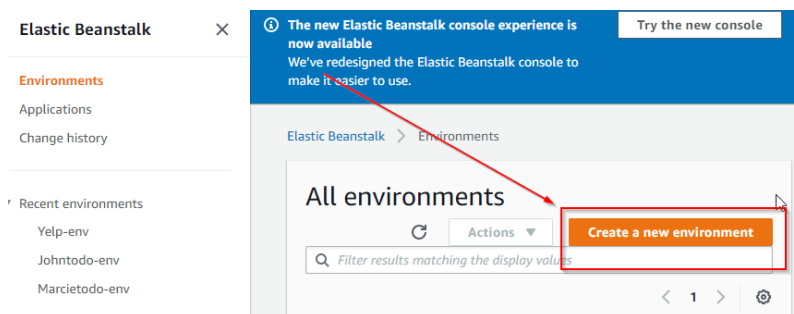2. Once logged in, you want to search and select the "Elastic Beanstalk" service.
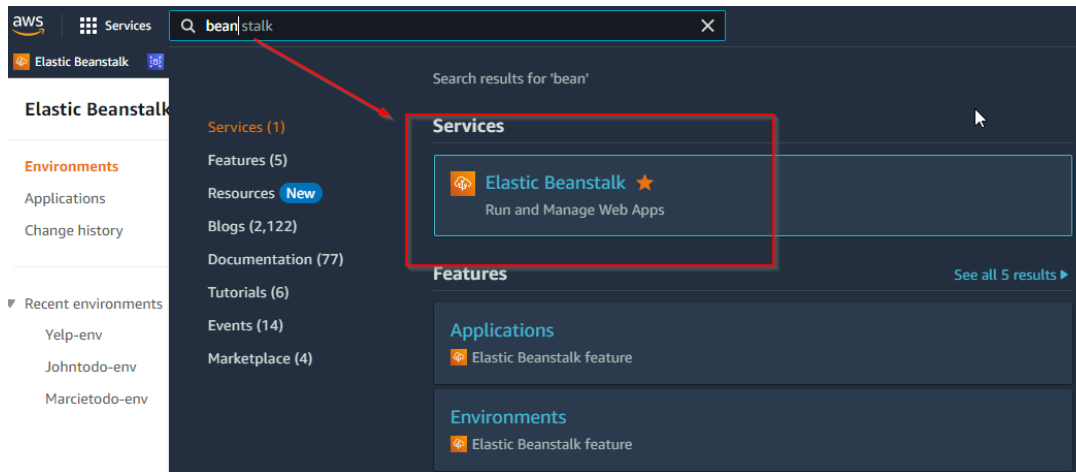
3. Select **"Create a new environment "**

4. Select **"Web Server Environment"**

Select environment tier

Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web servers are standard applications that listen for and then process HTTP requests, typically over port 80. Workers are specialized applications that have a background processing task that listens for messages on an Amazon SQS queue. Worker applications post those messages to your application by using HTTP.

○ Web server environment
  Run a website, web application, or web API that serves HTTP requests.
  Learn more ⬏

○ Worker environment
  Run a worker application that processes long-running workloads on demand or performs tasks on a schedule.
  Learn more ⬏

Cancel   **Select**

5. Give your application a name then scroll down and select **"Node.js"** as a platform
   Accept the other prefilled defaults for Node

Application information

Application name

```
rest_rant
```

Up to 100 Unicode characters, not including forward slash (/).

▶ **Application tags (optional)**

Platform

○ Managed platform
  Platforms published and maintained by
  Amazon Elastic Beanstalk. Learn more ⬏

○ Custom platform
  Platforms created and owned by you.

Platform

```
Node.js                                    ▼
```

Platform branch

```
Node.js 16 running on 64bit Amazon Linux 2        ▼
```

Platform version

```
5.6.4 (Recommended)                         ▼
```

6. Scroll down and select **"upload your code"**

7. At this point you must prepare your local code to upload.
   Zip up all the code shown here in your backend folder.
   You can name the zip whatever you like.

8. Now browse to select that new zip file of your backend code with the AWS GUI and select **"Create environment"**
   This may take several minutes to build out now.

9. Once it's complete don't worry about any status error in the console.
   You still have to setup your database and environment variables.

10. Now create Your AWS database
    Select Configuration and scroll down to **"Database"** and select **edit**

- Select **"Postgres"** as the engine
- Select **"db.t3.micro"** as the instance
- (Very important being if you don't select this instance type you may see some high charges on your AWS bill. This is the "free level" instance)
- Select a **username** and **password** for your AWS database
- Select **"delete"** as a detention policy. (again, to save on possible charges not included in the "free tier")
- Select **"Apply"** this will take some time to re-build again

11. Now you have to setup access to the database from any IP address
   - Click **Configuration** again and scroll down to the database section.
   - Click on the link to the new database **"Endpoint"**

Next you will see your list of databases. Click on the one you just created

   - Scroll down to you see **security groups** and click on the **"inbound"** one

   - In the **"actions"** drop down select **"Edit inbound rules"**

- Select **"add rule"**, add **"custom TCP"** set the **port** as needed, set it to **"Anywhere ipv4"** and select **"Save Rules"**

- Now search and go cack to the "bean stalk" settings.
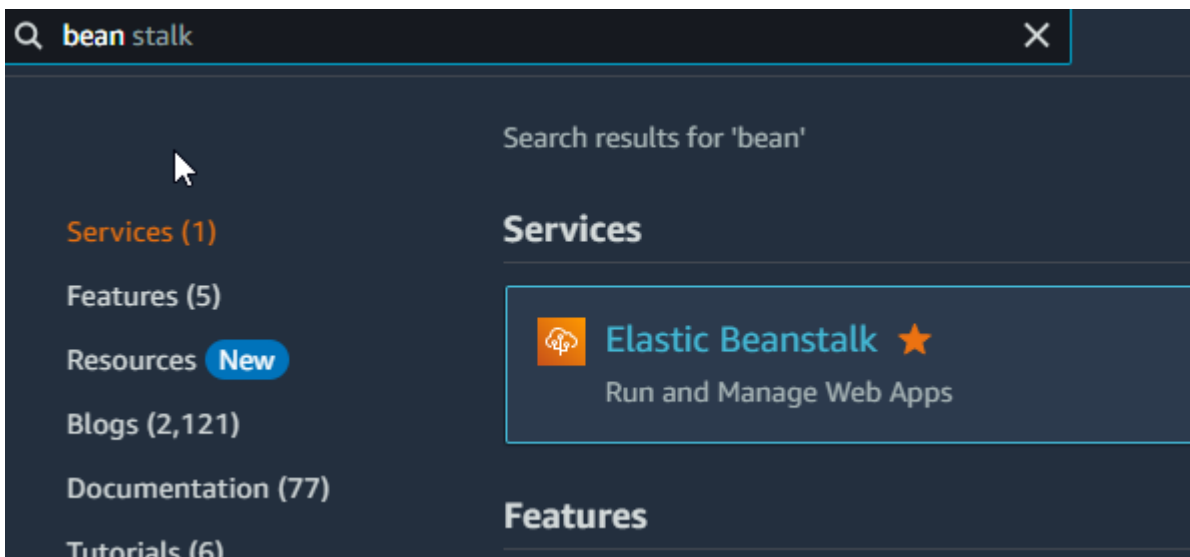
12. Time to add your system environment variables
    - First you need to copy that database endpoint that you created under **configuration** (You will need this link to enter as a variable but remove the :5432 port number from the end)

- Next go to **configuration** "software" and select **"edit"**

- Scroll down to Environment properties and enter your env variables for production
- It's important that you include that NODE_ENV=**production** setting as well as that long link from your newly created database. *Note that is case sensitive match what you used in your code*
- The other settings should match what you used previously when creating the DB on AWS
- The DB name on the AWS side (RDS_DB_NAME should be **"ebdb",** use that no matter what you named it locally for development

   This will now trigger another Beanstalk rebuild over several minutes
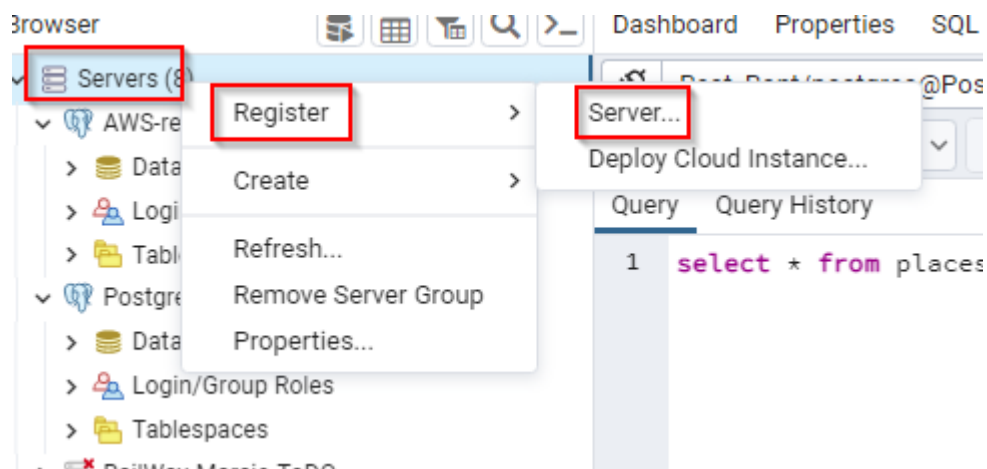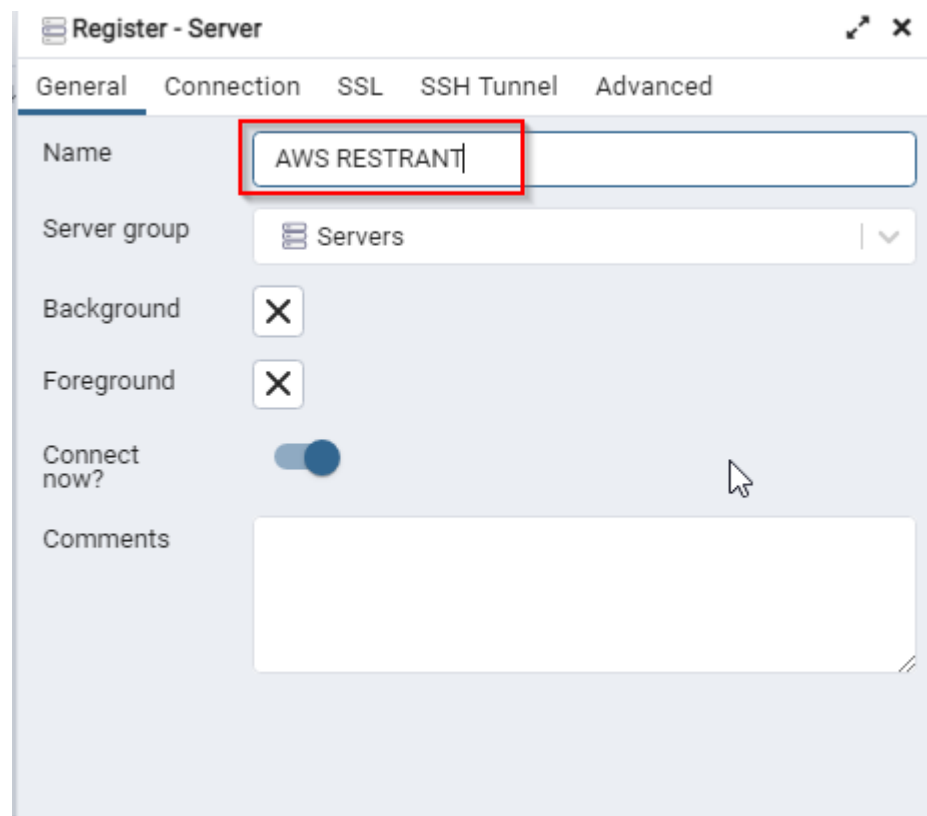
Setup the Database on AWS with your backup file

In the local PGADMIN tool connect to your AWS database

- First Right click on servers and select **"add new server"**

- Give the server a name

- Click **"Connection"** and enter The Server Details to connect remember to remove the port from the end again (:5432)

- Next right click on the newly connected server and select **"create database"**
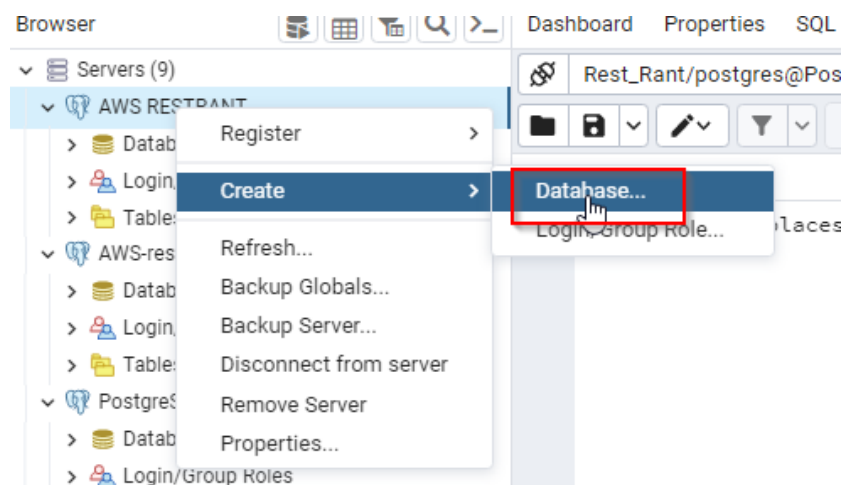
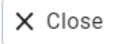- The database name in PGAdmin should match the AWS Environment entry of **"ebdb"**
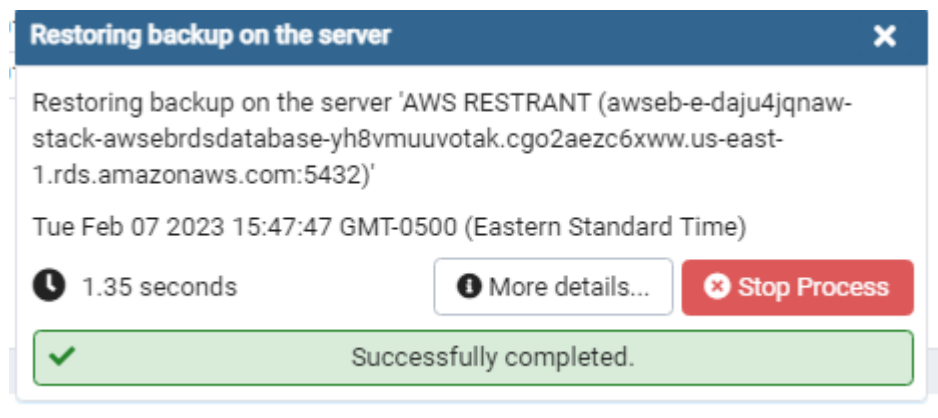
- Right Click on the newly created database and select **Restore**
- Browse to that same backup slq file we used earlier when creating the local database
-
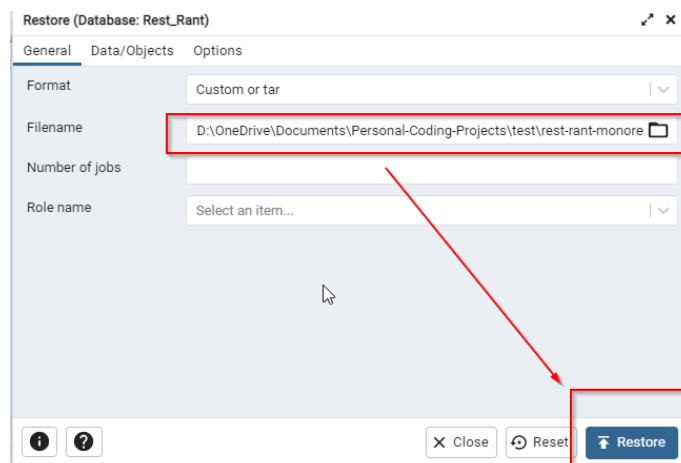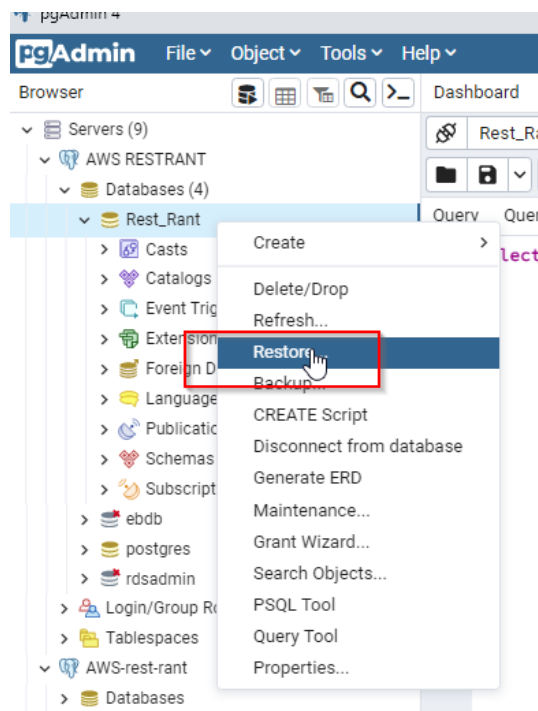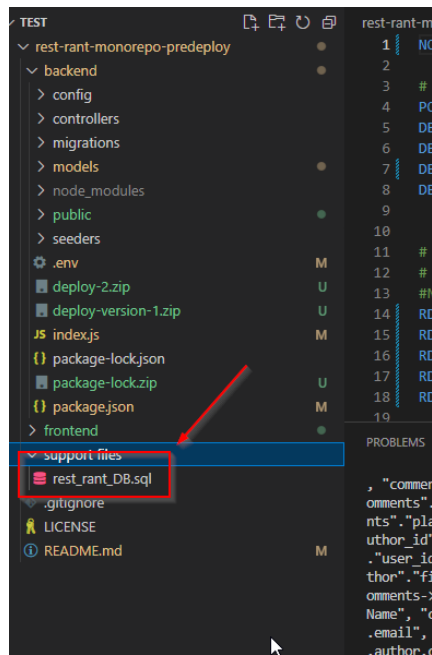
- You can go test your app now it should be ready for production

Click the link in AWS Beanstalk and test your app