



**Atividade 4 - Configurando e utilizando uma instância do PostgreSQL dentro do Amazon RDS.**

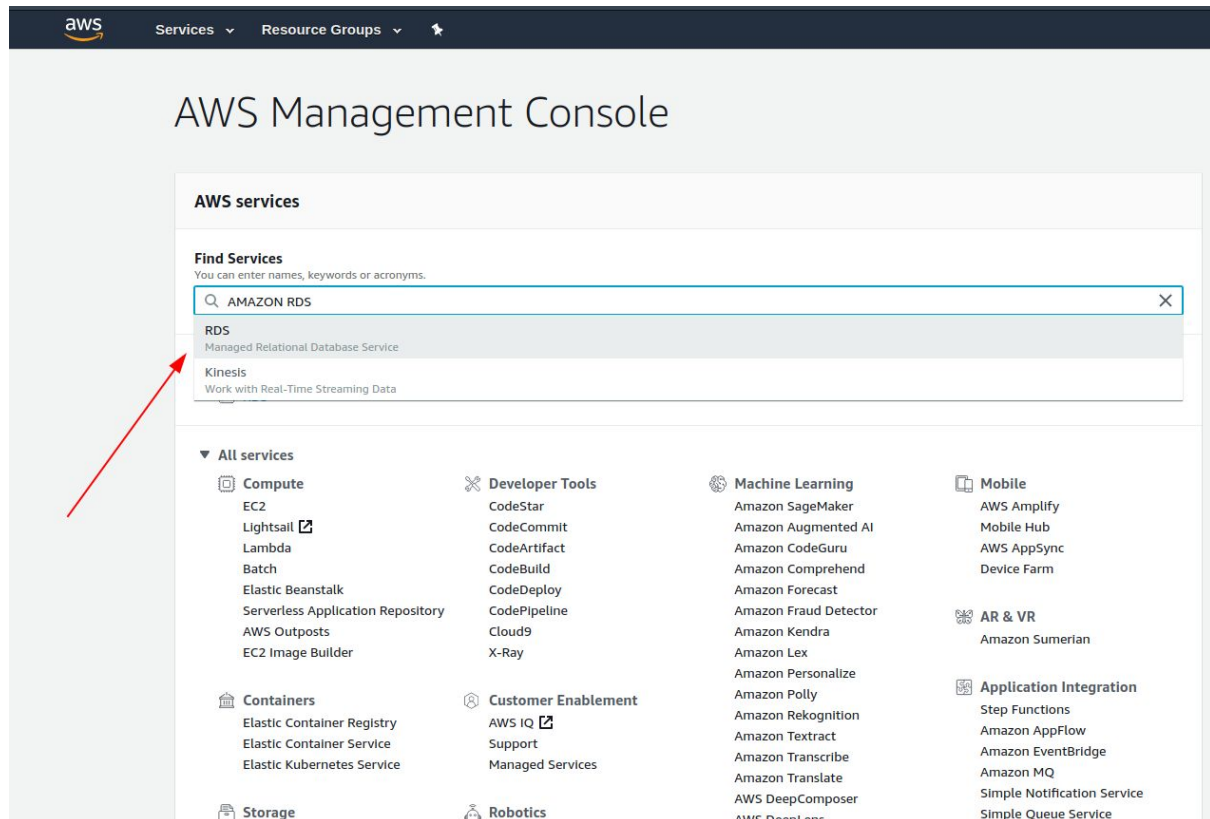
Disciplina : Desenvolvimento de Aplicações Distribuídas

Professor : Dr Flávio Sousa

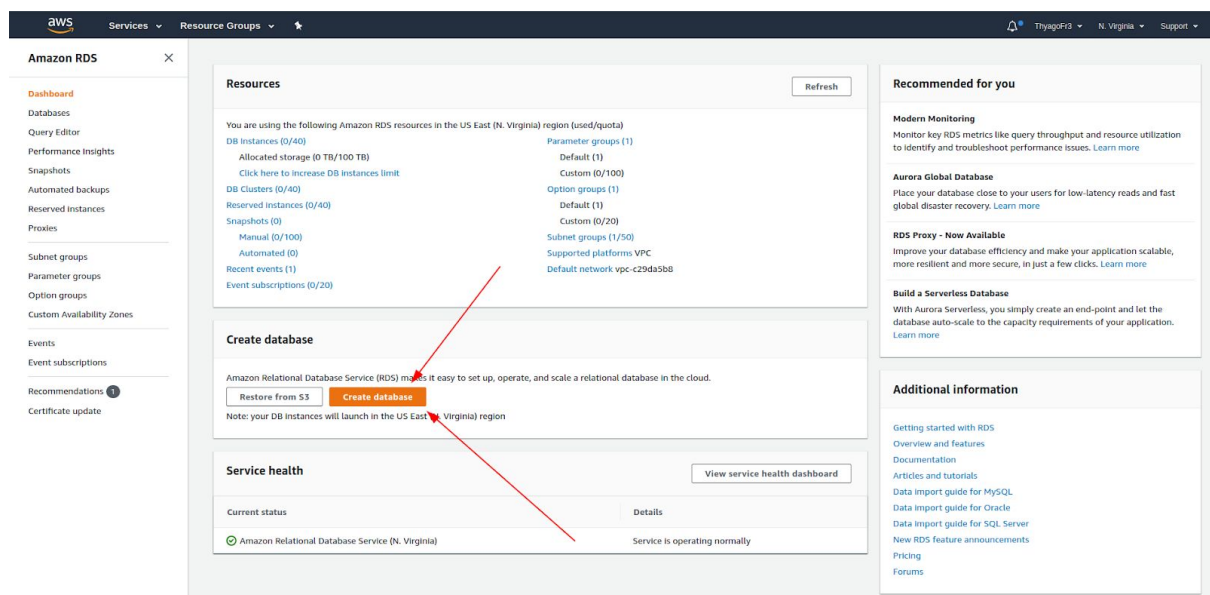
Alunos : Thyago Freitas da Silva - 392035

José Marcílio De Sousa - 385199

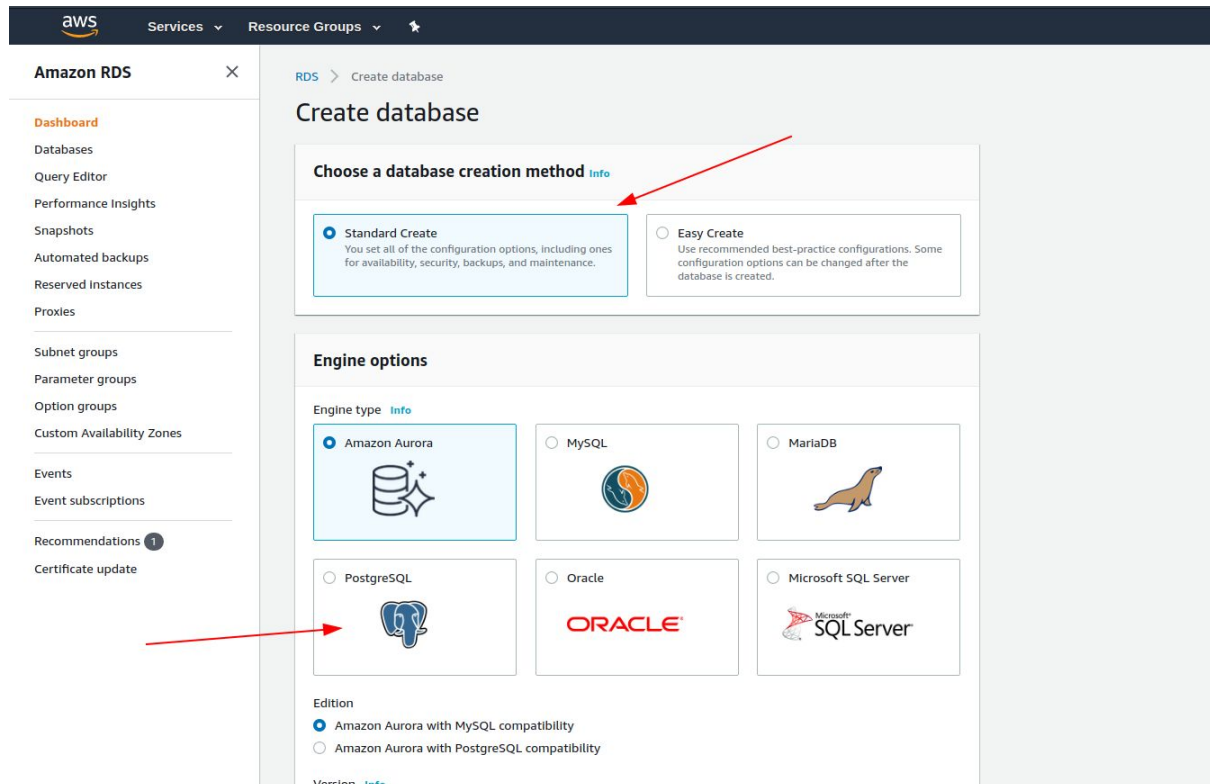
Passo 1 - Após logar no AWS Management Console, procurar por RDS na caixa de pesquisa. Clique em RDS.



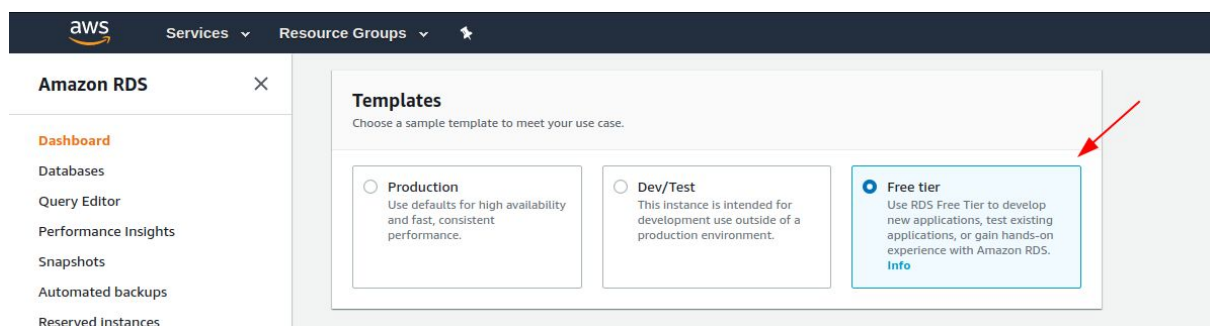
Passo 2 - Após carregar a página de dashboard Amazon RDS, buscar por “Create database”.



Passo 3 - Selecione a opção “Standard Create” para que possamos configurar alguns aspectos da instância. Nesse tutorial será utilizando o banco PostgreSQL 11-6.



Passo 5 - Na aba “Templates” escolha a opção “Free tier” para que possamos utilizar os recursos grátis disponibilizados pela AWS. Note as limitações desse template.



Passo 6 - Na aba “Settings” escolha o nome da instância do banco, usuário e senha(com no mínimo 8 caracteres).

The screenshot shows the 'Settings' tab in the AWS Management Console for an Amazon RDS instance. The left sidebar contains a navigation menu with options like Dashboard, Databases, Query Editor, Performance Insights, Snapshots, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom Availability Zones, Events, Event subscriptions, Recommendations (1), and Certificate update. The main content area is titled 'Settings' and includes the following sections:

- DB Instance Identifier**: A text input field containing 'godadDB'. A red arrow points to this field.
- Credentials Settings**:
  - Master username**: A text input field containing 'godad'. A red arrow points to this field.
  - Auto generate a password**: An unchecked checkbox.
  - Master password**: A password input field with masked characters. A red arrow points to this field.
  - Confirm password**: A password input field with masked characters. A red arrow points to this field.

Passo 7 - Na aba “DB Instance size” deixe o valor pré-selecionado. Esse valor é setado automaticamente quando escolhemos a opção “Free tier”.

The screenshot shows the 'DB instance size' tab in the AWS Management Console. The page displays the 'DB instance class' section with the following options:

- Standard classes (includes m classes)**: Unselected.
- Memory Optimized classes (includes r and x classes)**: Unselected.
- Burstable classes (includes t classes)**: Selected (indicated by a blue radio button).

Below the selected option, a dropdown menu shows the chosen instance class: **db.t2.micro**. The dropdown also displays '1 vCPUs', '1 GiB RAM', and 'Not EBS Optimized'. A red box highlights the 'Burstable classes' option and the dropdown menu. At the bottom, there is an unchecked checkbox for 'Include previous generation classes'.

Passo 8 - Na aba “Storage” deixe as opções pré-configuradas (note que no “Free tier” o banco por padrão terá 20GB de espaço e utilizará armazenamento SSD).

**Storage**

Storage type [Info](#)  
General Purpose (SSD)

Allocated storage  
20 GIB  
(Minimum: 20 GiB, Maximum: 16384 GiB) Higher allocated storage **may improve** IOPS performance.

Storage autoscaling [Info](#)  
Provides dynamic scaling support for your database's storage based on your application's needs.

☒ **Enable storage autoscaling**  
Enabling this feature will allow the storage to increase once the specified threshold is exceeded.

Maximum storage threshold [Info](#)  
Charges will apply when your database autoscales to the specified threshold  
1000 GIB  
Minimum: 21 GiB, Maximum: 16384 GiB

Passo 9 - Na aba “Connectivity” deixe o valor setado para a VPC e clique em “Additional connectivity configuration” para que possamos tornar o banco acessível pela internet e configurar o tipo de autenticação de acesso ao mesmo.

**Connectivity**

Virtual private cloud (VPC) [Info](#)  
VPC that defines the virtual networking environment for this DB instance.  
Default VPC (vpc-c29da5b8)  
Only VPCs with a corresponding DB subnet group are listed.

After a database is created, you can't change the VPC selection.

► **Additional connectivity configuration**

Após clicar em “Additional connectivity configuration” você poderá ver uma tabela de configuração como na tabela abaixo. Mantenha o valor selecionando automaticamente em “Subnet group”. Certifique-se de quem em “Publicly accessible” a opção selecionada é a “Yes”, pois essa opção garante que dispositivos dentro e fora da VPC possam acessar a instância do banco de dados. Escolha uma zona entre as listadas como acessíveis e anote a porta que será utilizada para acesso ao banco( 5432 é a porta padrão do PostgreSQL);

Query Editor

Performance Insights

Snapshots

Automated backups

Reserved instances

Proxies

Subnet groups

Parameter groups

Option groups

Custom Availability Zones

Events

Event subscriptions

Recommendations 1

Certificate update

▼ Additional connectivity configuration

**Subnet group** [Info](#)

DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you selected.

default-vpc-c29da5b8 ▼

**Publicly accessible** [Info](#)

☒ Yes

Amazon EC2 instances and devices outside the VPC can connect to your database. Choose one or more VPC security groups that specify which EC2 instances and devices inside the VPC can connect to the database.

☐ No

RDS will not assign a public IP address to the database. Only Amazon EC2 instances and devices inside the VPC can connect to your database.

**Existing VPC security groups**

Choose VPC security groups ▼

default X

**Availability Zone** [Info](#)

us-east-1a ▼

**Database port** [Info](#)

TCP/IP port that the database will use for application connections.

5432

Passo 10 - Na aba “Database authentication” utilizaremos a senha e usuário que configuramos no passo 6, para isso selecione a opção “Password authentication”. Note que existem outras possibilidades de autenticação, como a IAM onde você terá que criar um usuário utilizando o serviço de gerência de acesso da própria AWS. Logo após clique em “Create database”.

**Database authentication**

**Database authentication options** [Info](#)

☒ Password authentication

Authenticates using database passwords.

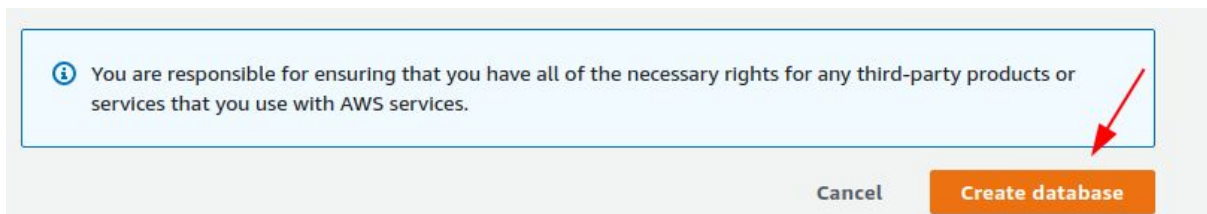
☐ Password and IAM database authentication

Authenticates using the database password and user credentials through AWS IAM users and roles.

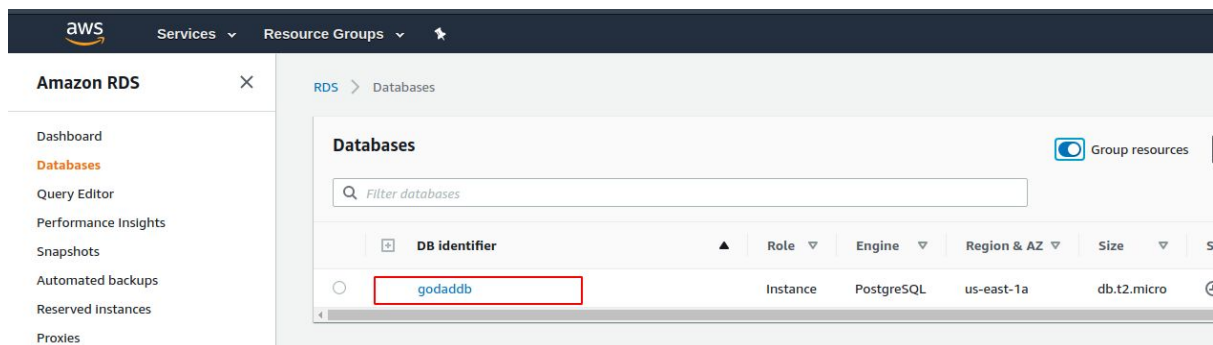
☐ Password and Kerberos authentication

Choose a directory in which you want to allow authorized users to authenticate with this DB instance using Kerberos Authentication.

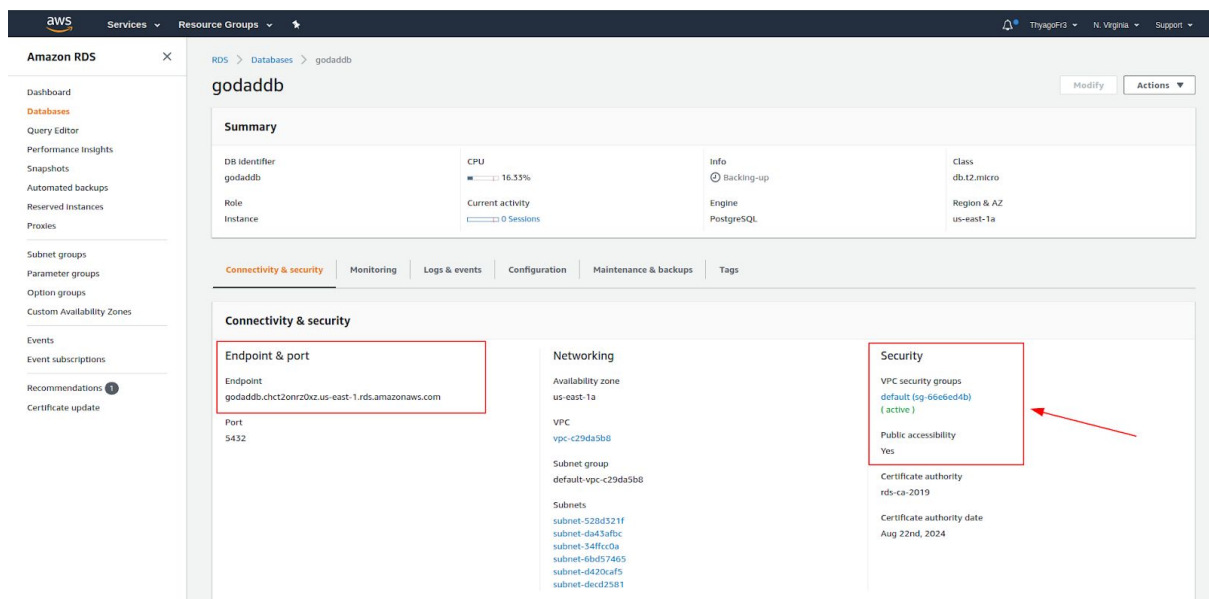




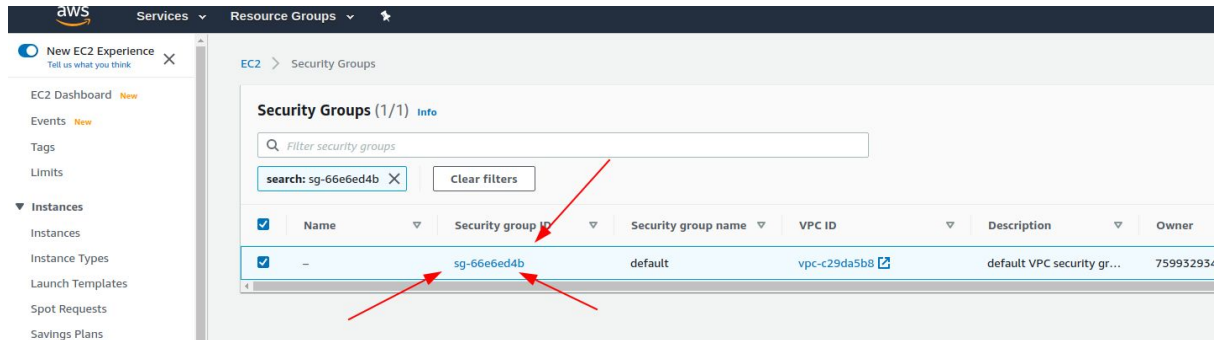
Passo 11 - Após clicar em “Create database” você será redirecionado para o dashboard do Amazon RDS. A criação da instância pode demorar alguns minutos. Quando aparecer a notificação de que a instância foi criada, selecione a mesma clicando em cima do seu nome.



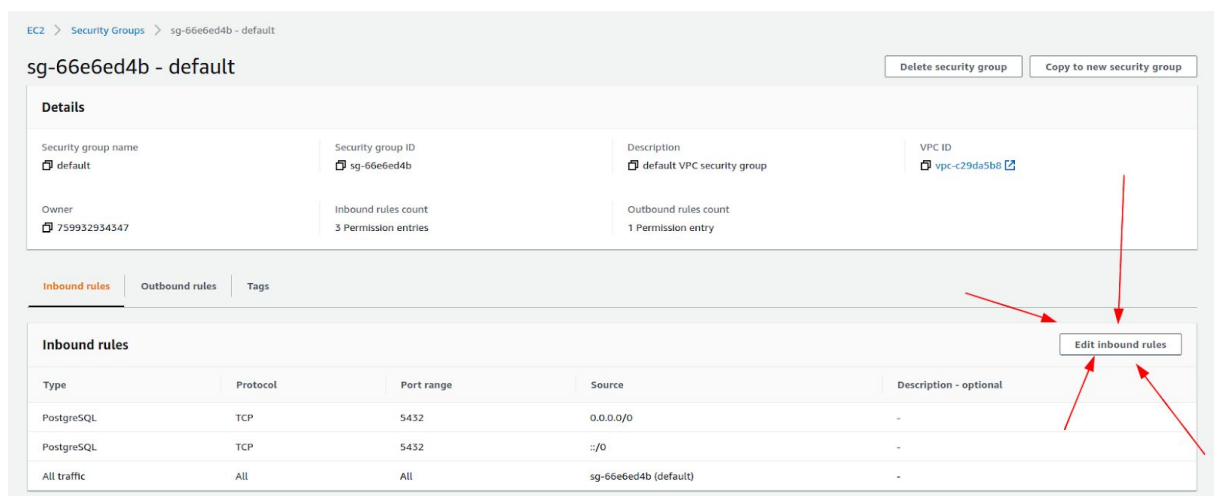
Passo 12 - Note que agora já temos um endpoint que iremos utilizar para se conectar a instância criada do Amazon RDS. Porém, precisamos primeiro expor a porta 5432 (padrão de acesso do PostgreSQL). Para isso clique no link na parte “Security” na aba “Connectivity & security”. Observação : Anote o endpoint para que possa ser utilizado depois.



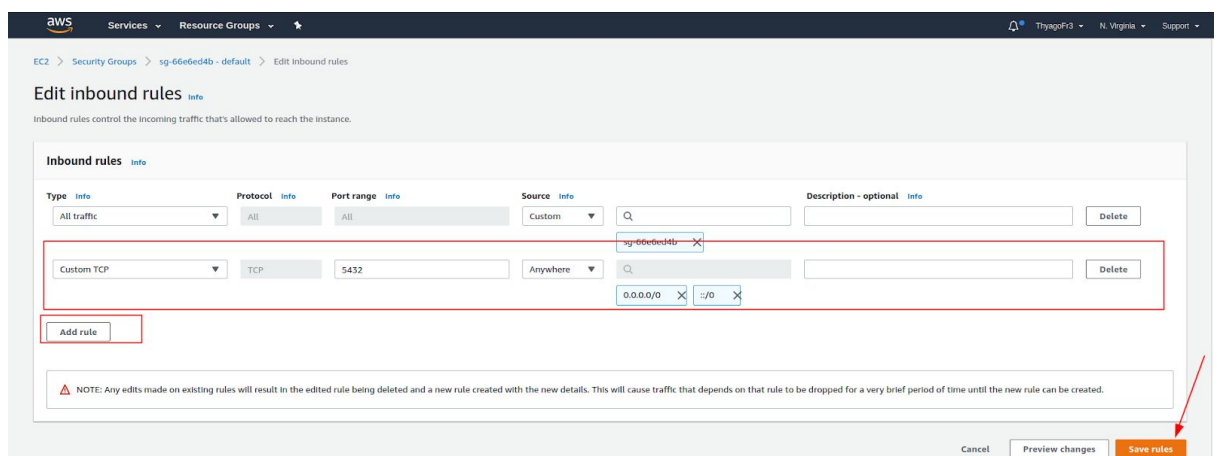
Passo 13 - No dashboard que é aberto, clique no “Security group ID” que foi criado para você, é lá que iremos configurar a exposição da porta 5432.



Passo 14 - Após concluir o carregamento da nova página, clique na opção “Edit inbound rules”.

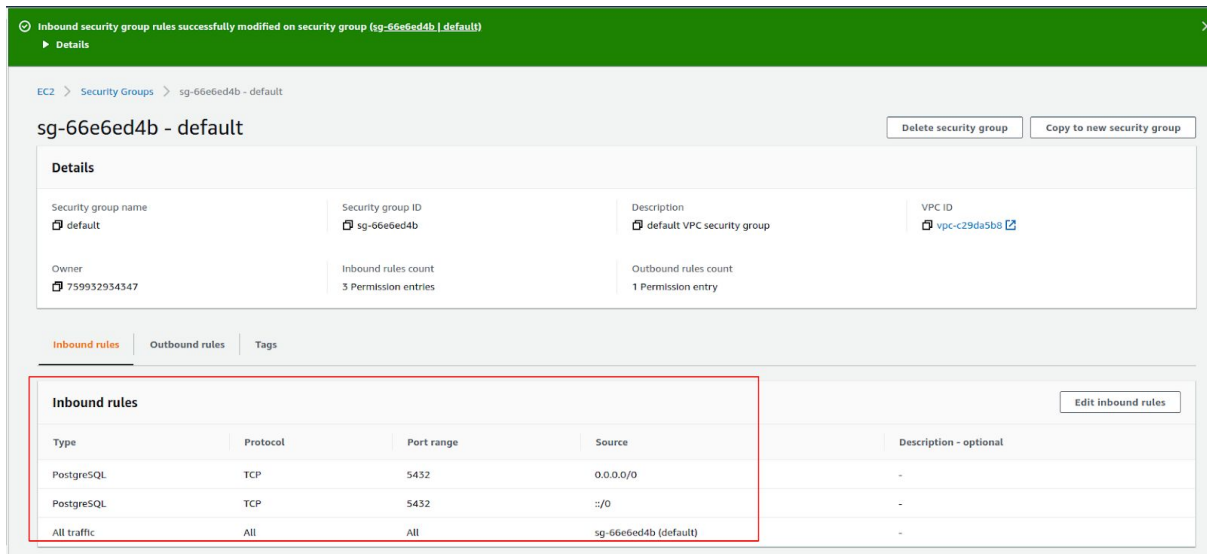


Passo 15 - Clique em “Add rule”, selecione “Custom TCP” e selecione a porta “5432”. Na coluna “Source” selecione a opção “Anywhere” para que o acesso a essa porta possa ser feita por toda a internet. Clique em “Save rules”.





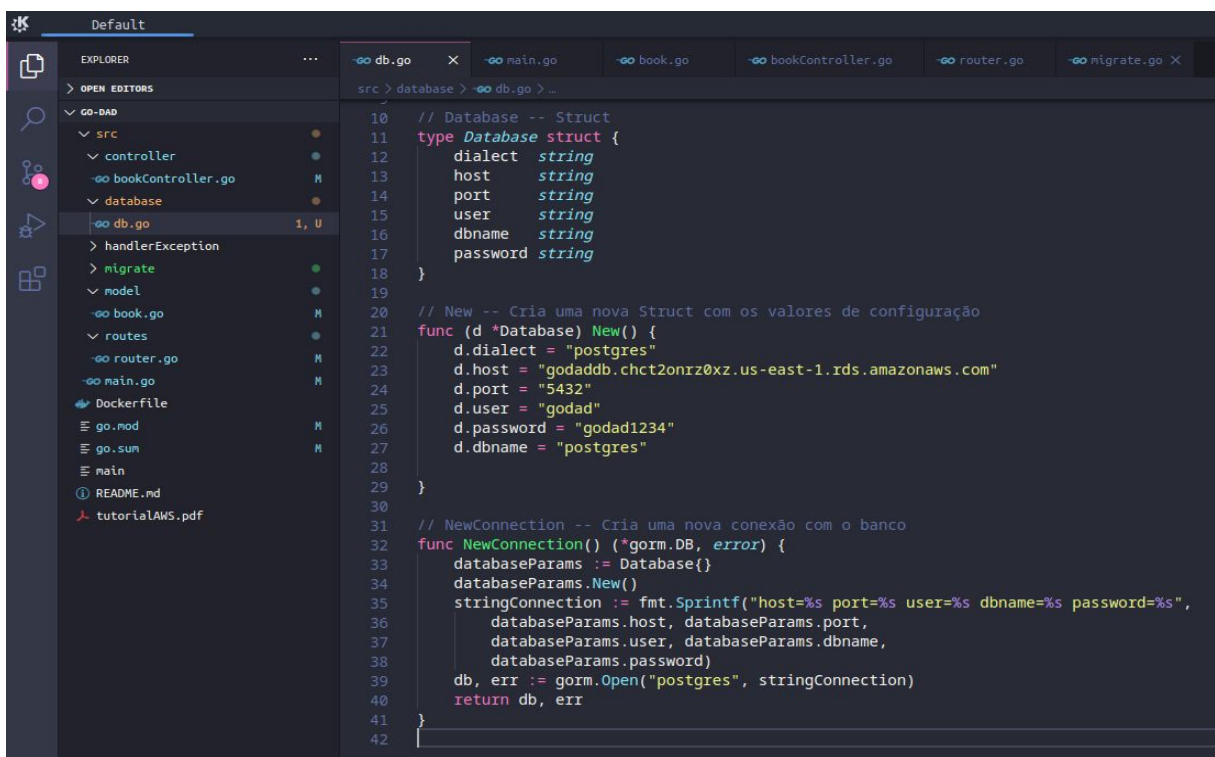
Após o passo 15 você poderá ver que a nova regra de acesso foi adicionada. Finalmente poderemos se conectar ao PostgreSQL.



Como desenvolvemos uma API simples na atividade 2 da disciplina, optamos por utilizar o banco para salvar os dados que antes eram salvos em memória. O projeto pode ser encontrado [aqui](#). A API segue o padrão REST e possui 4 rotas :

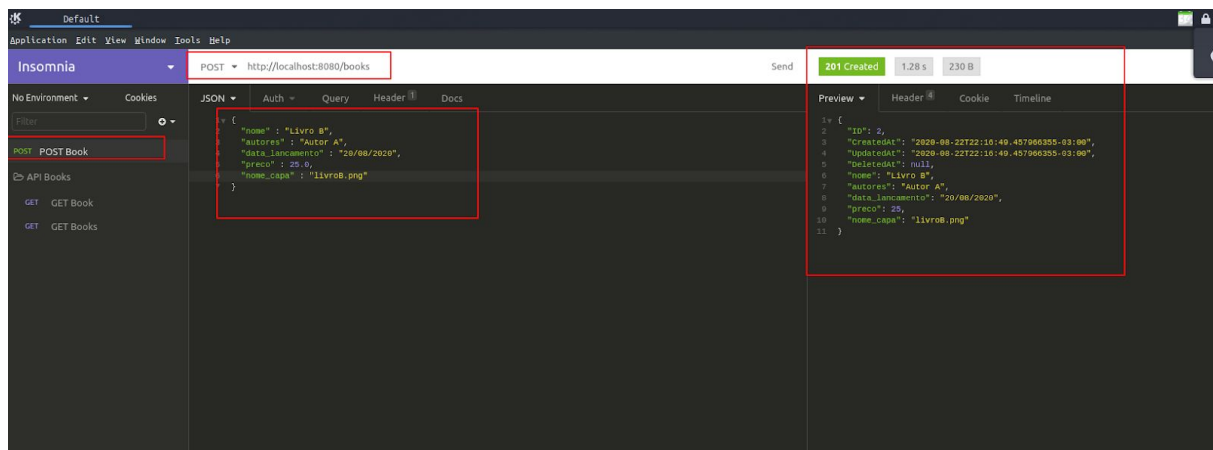
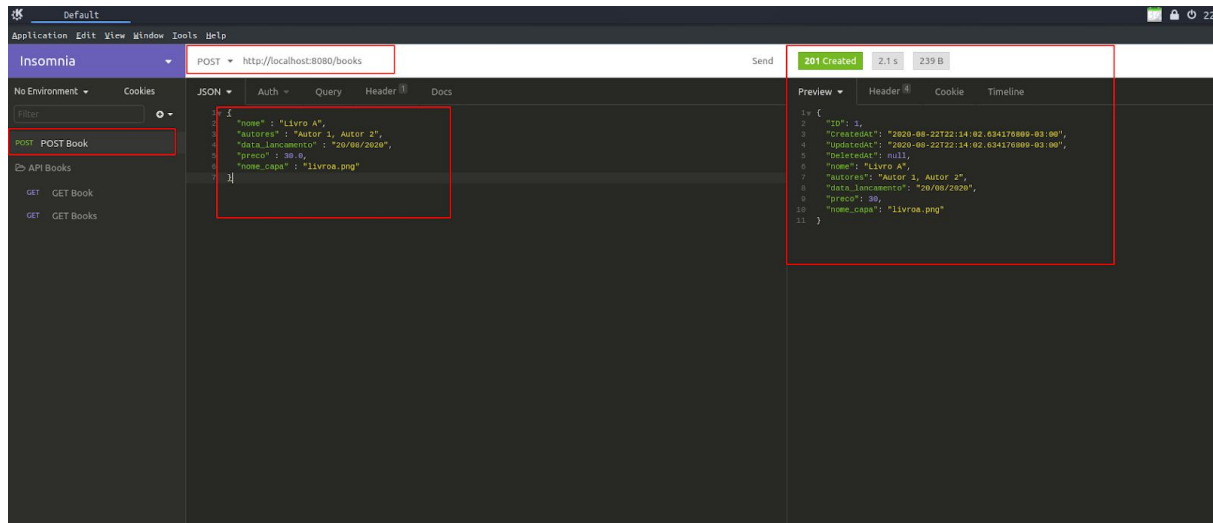
- `http://ip:porta/books` - GET (Buscar todos os livros cadastrados)
- `http://ip:porta/books/id` - GET (Buscar livro com ID especificado)
- `http://ip:porta/books/id` - DELETE (Deletar livro com ID especificado)
- `http://ip:porta/books` - POST (Cadastrar livro)

Abaixo segue um trecho do código em go utilizado para gerar conexões ao banco POSTGRESQL rodando no AMAZON RDS.

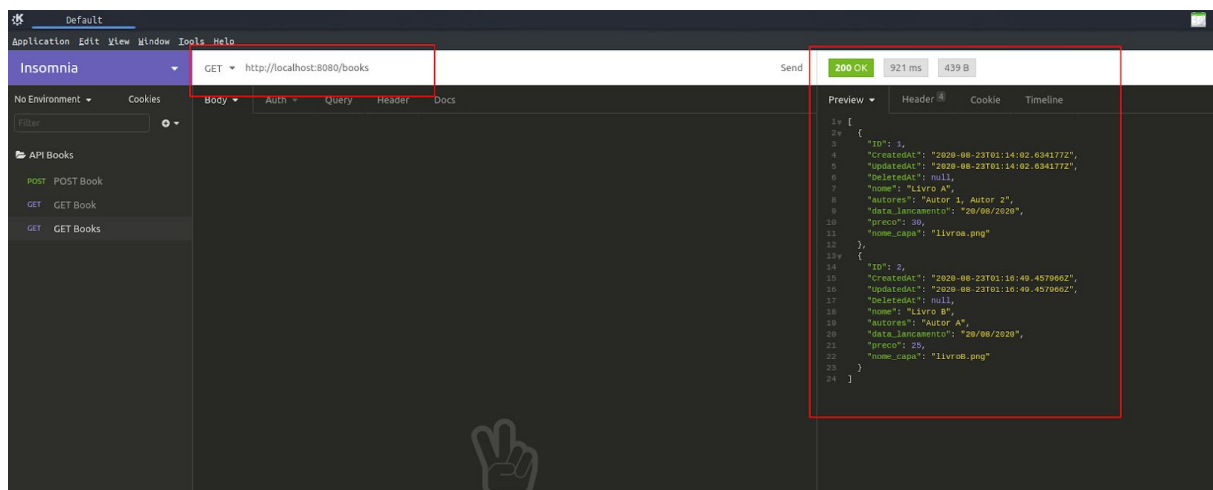


Abaixo temos alguns testes da API já utilizando o banco de dados como sistema de persistência.

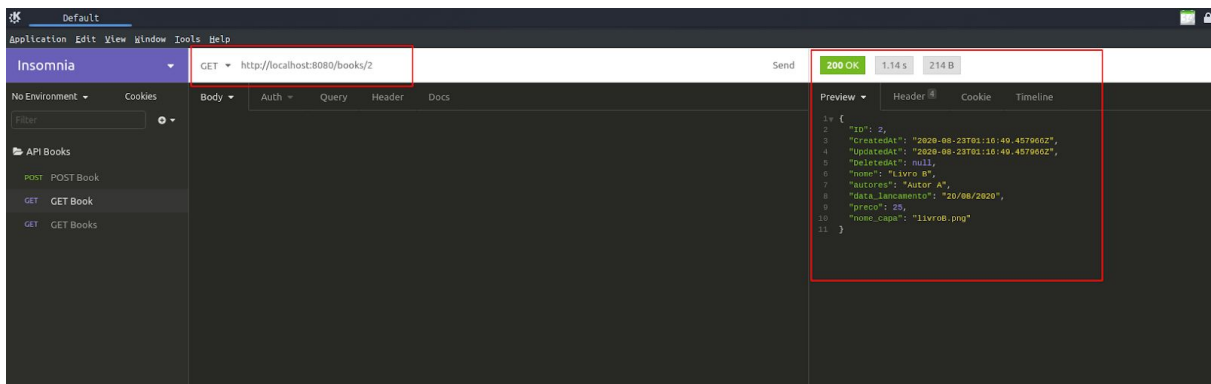
- Cadastro de novos livros.



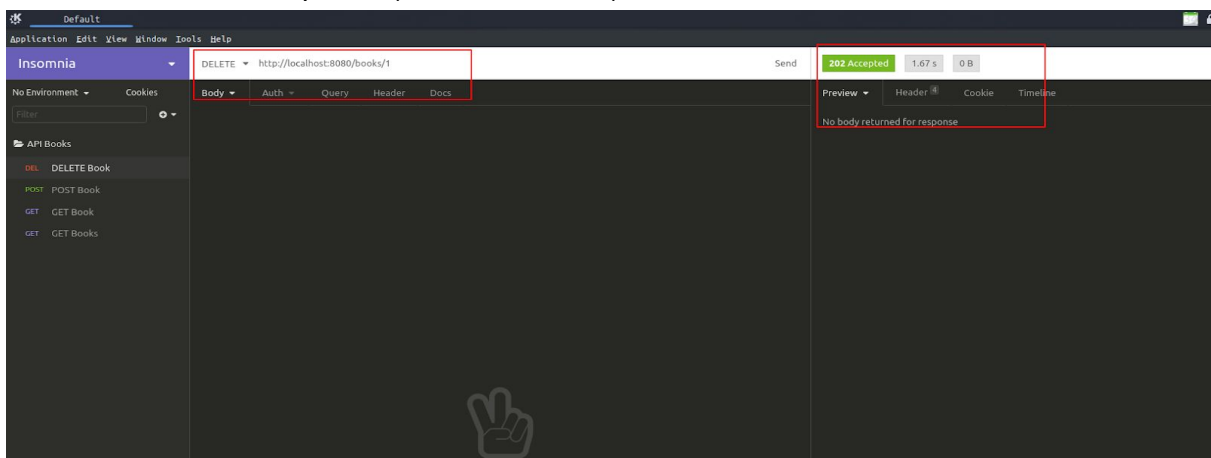
- Listando todos os livros cadastrados.



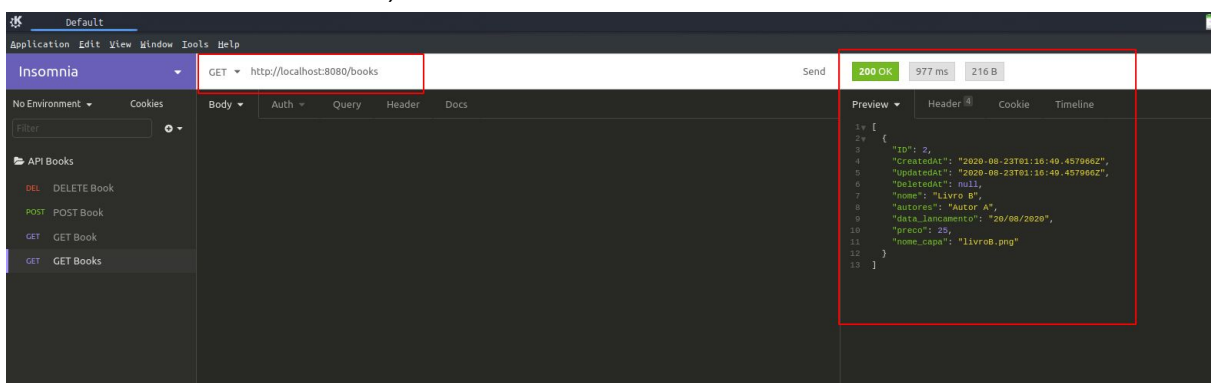
- Buscando livro por um ID específico.



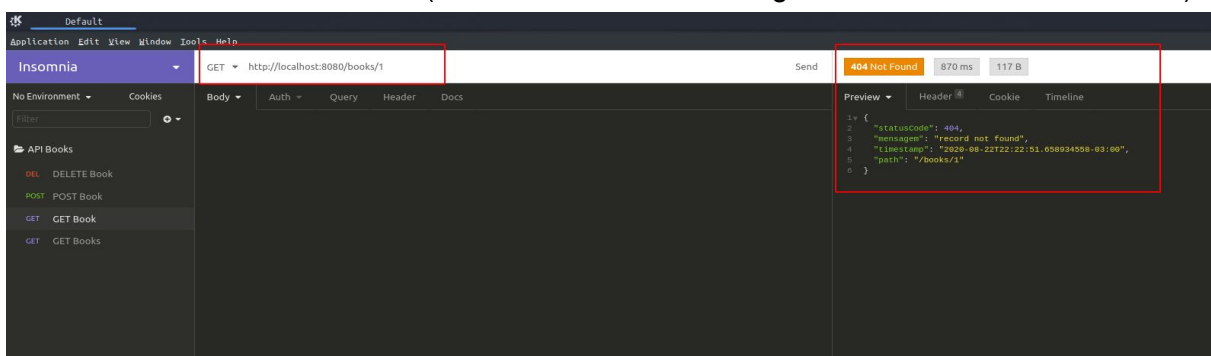
- Deletando livro por ID (nesse caso ID 1).



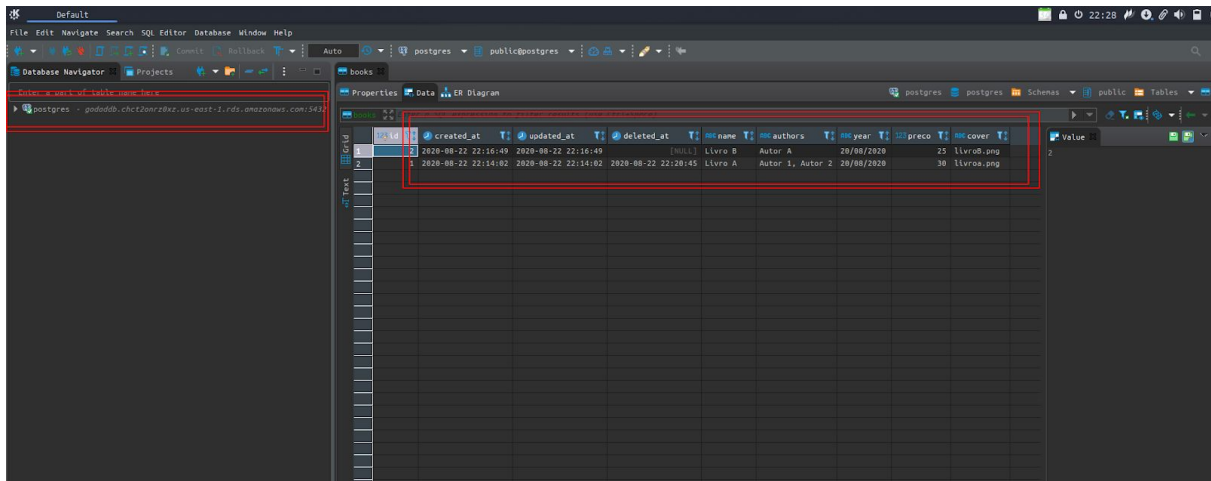
- Listando todos os livros cadastrados (aqui podemos ver que o livro com ID 1 foi excluído com sucesso) .



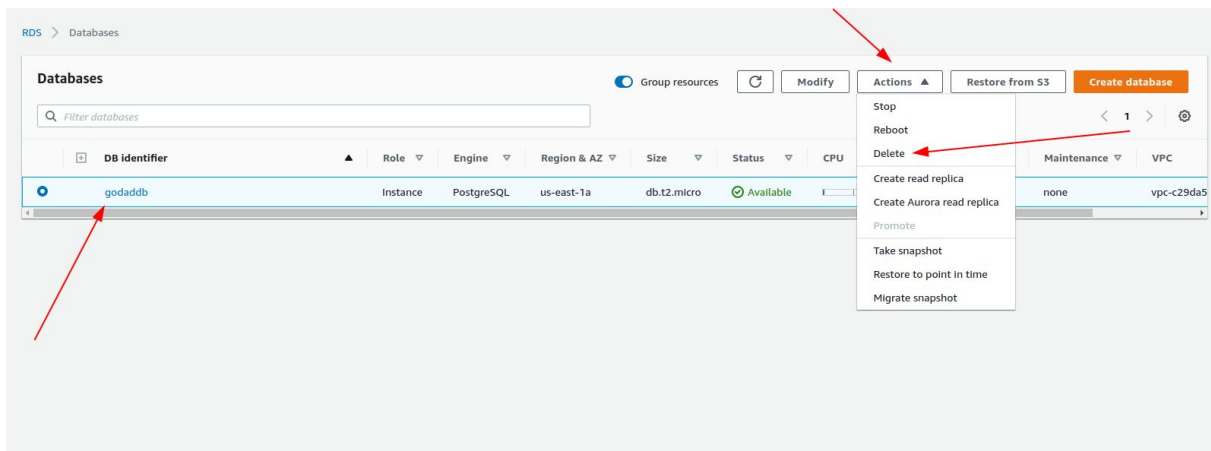
- Buscando livro com ID 1 (excluído anteriormente, logo HTTP status é NOT FOUND).



Segue abaixo um print da tela do DBeaver para visualizar o banco POSTGRESQL. Como podemos ver a tabela foi criada no banco e os dados foram inseridos e removidos com sucesso da mesma pela API.



Passo 16 - Para finalizar, basta excluir a instância do banco indo no dashboard de gerência do Amazon RDS, selecionar a instância, clicar no botão “Actions” e selecionar a opção “Delete”.



Logo após clicar, irá aparecer um modal de confirmação. Basta digitar “delete me” e clicar em delete.

