Sumário

[Introdução 2](#_Toc100522734)

[Objetivo 2](#_Toc100522735)

[Step 1 3](#_Toc100522736)

[Criação do Compartment 3](#_Toc100522737)

[Criação de Usuario 3](#_Toc100522738)

[Criação de Grupo 4](#_Toc100522739)

[Criação de Policies 5](#_Toc100522740)

[Criação de SSH-KEY 6](#_Toc100522741)

[Criação de Instancia VM 7](#_Toc100522742)

[Step 2 8](#_Toc100522743)

[Entrar na Instancia via shell 8](#_Toc100522744)

[Instalar Docker 8](#_Toc100522745)

[Instalar Docker-Compose 9](#_Toc100522746)

[Instalar Git 9](#_Toc100522747)

[Step 3 10](#_Toc100522748)

[Criação do Docker-Compose-Services 10](#_Toc100522749)

# Introdução

Nesse projeto, utilizarei algumas ferramentas contidas na OCI (Oracle Claud Infraestructure) visto que, recentemente passei na certificação “Oracle Cloud Infrastructure Foundations 2021 Associate“.

Utilizarei também:

- Python com framework Flash para criação de API Gateway;

- Persistência de dados em banco relacional com PostgreeSQL;

- Camada de cache em memória com Redis;

- Utilização de um ORM para manipulação dos dados;

- Docker para criação de containers de microsserviços.

Vamos começar...

## Objetivo

Criar duas aplicações básicas de microserviços:

O primeiro deles deverá ser um cadastro de usuários, contendo as seguintes informações:

- create\_user, show\_all\_user, show\_one\_user, edit\_user, edit\_password e delete\_user

Tabela de usuários “users” deverá conter os campos: user\_id, nick\_name, nick\_name, \_name, \_name, email, phone\_number, phone\_number, updated\_at.

O segundo será um serviço de OS (ordem de serviço) que deverá conter o user\_id contido no banco de dados. Deverá ter as seguintes informações:

- order\_id, user\_id, item\_description, item\_quantity, item\_price, total\_value, created\_at, updated\_at.

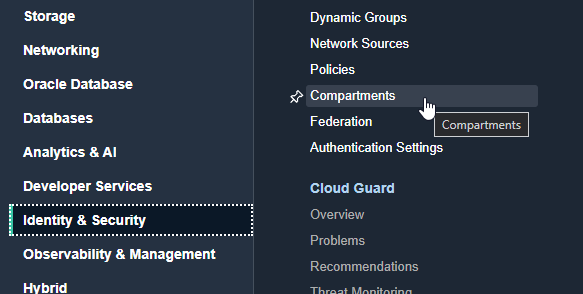
# Step 1

Nesse primeiro passo, configurei nosso ambiente OCI para rodar toda a aplicação em cloud.

## Criação do Compartment

Iniciei criando um *conpartiment.*

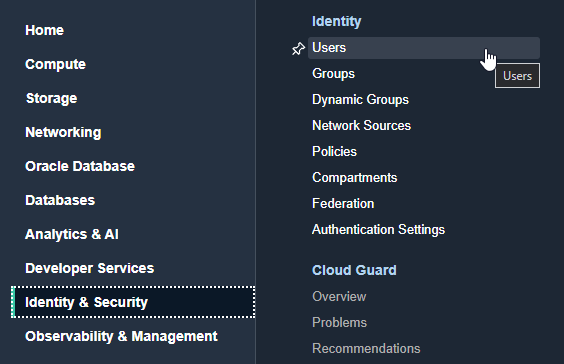
No ícone de hamburguer na página inicial do OCI, em *> Identity & Security > compartments* criei um compartimento com o nome de **Developmet.**

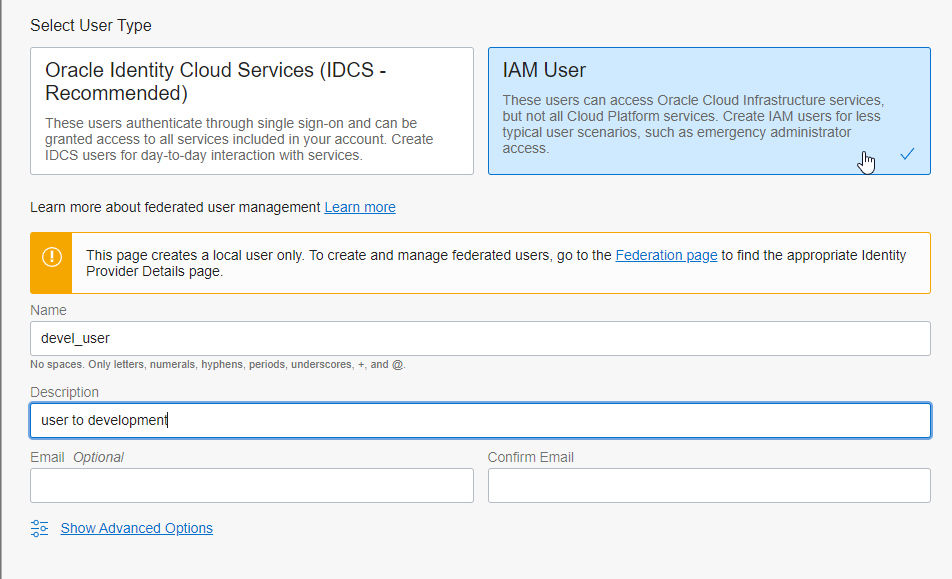


## Criação de Usuário

Após isso, criei um usuário de desenvolvimento para ter acesso somente ao compartimento **Developmet**.

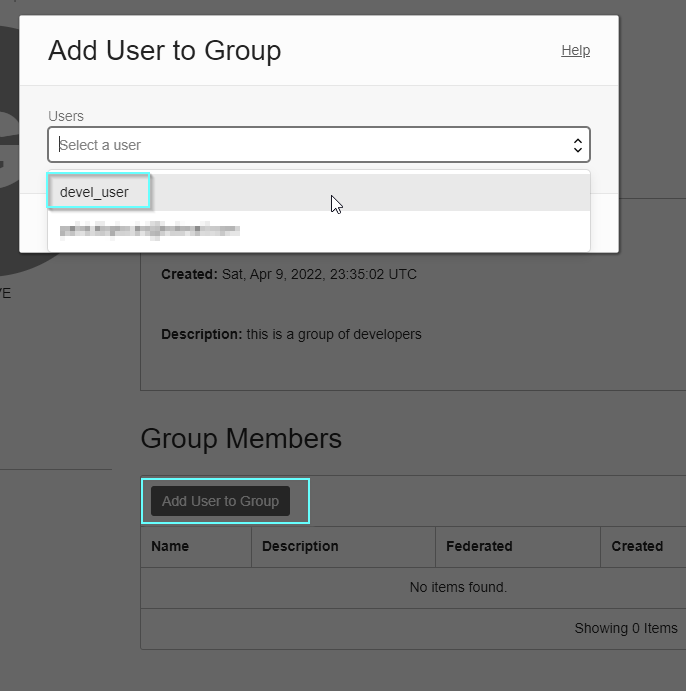
Novamente no menu lateral, em *> Identity & Security > users*, criei o usuário **devel\_user,** com o tipo IAM USER. Esses usuários podem acessar os serviços do Oracle Cloud Infrastructure, mas nem todos os serviços do Cloud Platform. Os usuários do IAM são cenários de usuário atípico, como acesso de administrador de emergência.





## Criação de Grupo

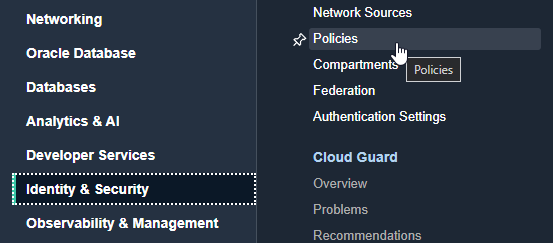
No menu lateral, em *> Identity & Security > group*s, criei um grupo com o nome de **Developer\_Group.**  Depois disso, adicionei o usuário **devel\_user** ao grupo clicando no botão “Add user to Group”.



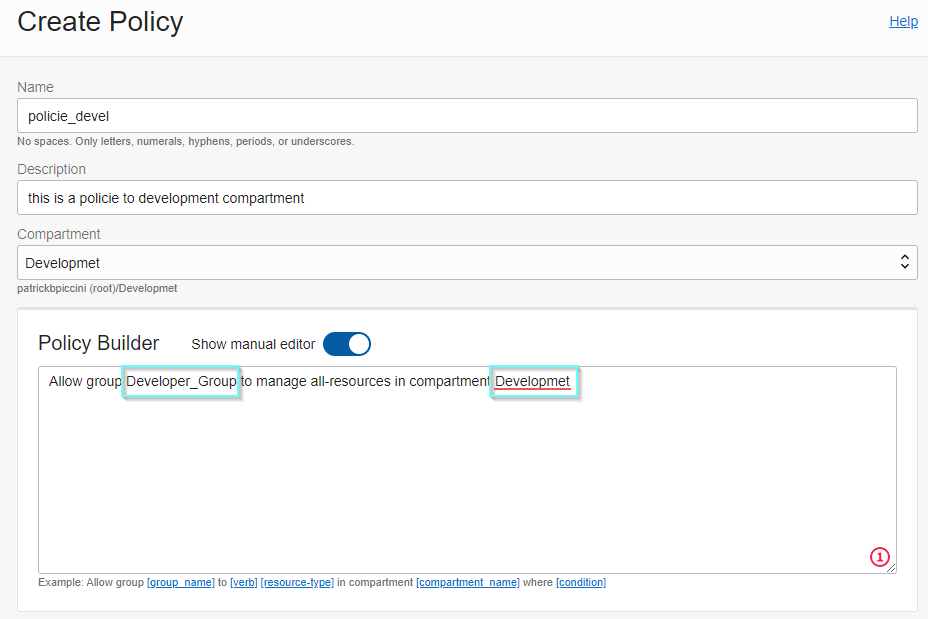
## Criação de Policies

Finalizando o processo de criação de Usuário, Grupo, e adição do usuário ao grupo, é necessário atribuir uma sequência de política de permissões ao grupo, para que assim, o grupo criado tenha acesso ao compartimento *Developer* para fazer as devidas interações*.*

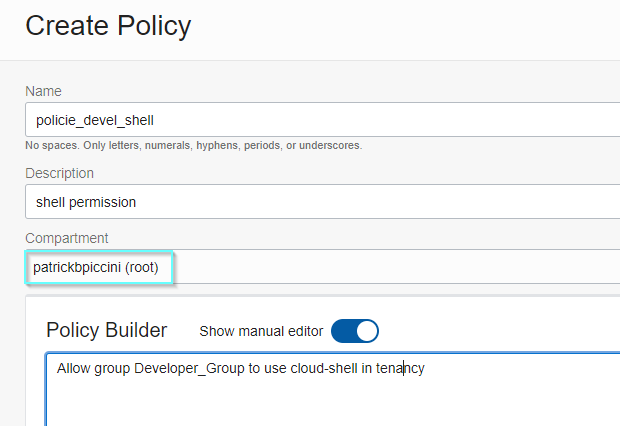
Vamos em *> Identity & Security > Policies*



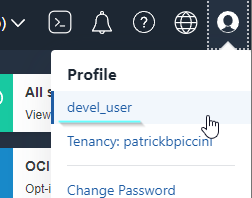
Após ser direcionado a página de policies, selecionei o compartimento Developmet para receber a política que criei. Na policie, permiti o Developer\_Group (e todos seus usuários) a utilização de todos os recursos no compartimento OCI.



Também será criado uma outra policie no compartimento **root**, para liberar o acesso ao terminal cloud shell, através dos usuários do grupo *Developmet.* Isso deverá ser feito devido ao fato de que precisarei de uma ssh-key disponibilizada pelo usuário, para conseguir criar nossa instancia no OCI.

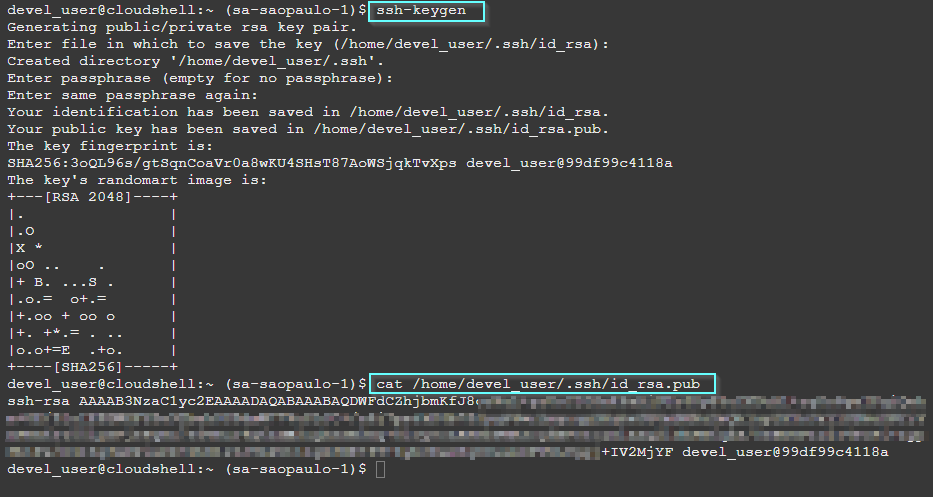


## Criação de SSH-KEY

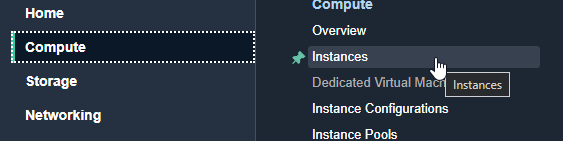
**Logado no usuário *devel\_user,* conectei ao Oracle Cloud Shell para adquirir uma ssh\_key que posteriormente irei utilizar. Para isso, usei alguns simples comandos para a criação dessa chave.

ssh-keygen – Para criação da chave

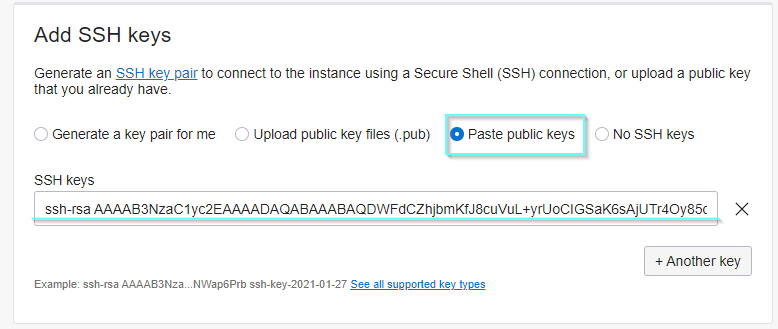
cat /home/your\_user/.ssh/id\_rsa.pub – Mostrará o conteúdo contido no arquivo id\_rsa.pub, que foi criado com o comando anterior



## Criação de Instancia VM

Ainda conectado ao usuário *devel\_user* criei uma instancia, que nada mais é que uma Virtual Macine. Em > *compute > instances > create instance.*

Ao criar uma instancia na página de configuração da VM, colei a ssh-key adquirida anteriormente no campo onde solicita essa chave. Isso é feito para que se consiga acessar a VM remotamente. As imagens do Oracle Linux, CentOS ou Ubuntu usam esse par de chaves SSH ao contrário de uma senha para autenticar um usuário.



OBS: A configuração da instancia que irei utilizar, são disponibilizadas pelo serviço *Oracle Cloud – Free Tier,* sendo ela 1 VMs de computação baseadas em AMD com 1/8 OCPU\*\* e 1 GB de memória cada.

Para mais informações dos serviços Free Tier, acesse:

<https://www.oracle.com/br/cloud/free/>

# Step 2

Após Configurar a infraestrutura no Oracle Cloud, necessita-se fazer algumas instalações de alguns programas que utilizarei. O Docker para a criação dos nossos containers de servidores e microsserviços, o Docker-Compose para facilitar a criação dos containers, e o Git, para versionamento de vocigo

Como desenvolvi os códigos fora de nossa VM da OCI, o Git é o fator principal para que todos os códigos sejam disponíveis facilmente.

## Entrar na Instancia via shell

Logado no usuário **devel\_user**, iniciei o cloud shell para conectar-se ao VM criada através do SSH Connection.

\* Detalhe, essa conexão pode ser feita de qualquer terminal ou comprador que tenha acesso ao SSH Connection, desde que se tenha cadastrado na hora da criação da VM a SSH-KEY da máquina que onde se conectará.

Com o comando ssh opc@<ip\_public> entrei na máquina e comecei a fazer a instalação com o pacote de instalação Dandified YUM (DNF) no Oracle Linux 8.

## Instalar Docker

Com a sequência de comando abaixo, foi instalado o Docker na VM:

> dnf install -y dnf-utils zip unzip

> dnf config-manager --add-repo=https://download.docker.com/linux/centos/docker-ce.repo

> dnf remove -y runc

> dnf install -y docker-ce --nobest

> systemctl enable docker.service

> systemctl start docker.service

> systemctl status docker.service

> docker version

Adicionar o usuário ao grupo docker para poder executar comandos Docker;

> sudo usermod -aG docker opc

> newgrp docker

Vemos que Docker está rodando e pronto para ser usando.

Texto

Descrição gerada automaticamente

## Instalar Docker-Compose

Foi instalado o Docker-Compose na VM com os seguintes comandos:

> sudo dnf -y install curl

> curl -s https://api.github.com/repos/docker/compose/releases/latest|grep browser\_download\_url|grep docker-compose-linux-x86\_64|cut -d '"' -f 4|wget -qi –

> ls -1 docker-compose-linux-x86\_64\*

> sha256sum -c docker-compose-linux-x86\_64.sha256

> chmod +x docker-compose-linux-x86\_64

> sudo mv docker-compose-linux-x86\_64 /usr/local/bin/docker-compose

> docker-compose version

Docker-Compose instalado com sucesso.

Tela preta com letras brancas

Descrição gerada automaticamente

## Instalar Git

Comandos para instalação do Git:

> yum install curl-devel expat-devel gettext-devel openssl-devel zlib-devel -y

> yum install gcc perl-ExtUtils-MakeMaker -y

> cd /usr/local/

> wget https://mirrors.edge.kernel.org/pub/software/scm/git/git-2.35.1.tar.gz

> tar zxvf git-2.35.1.tar.gz

> cd git-2.35.1/

> make prefix=/usr/local/git all

> make prefix=/usr/local/git install

> echo "export PATH=$PATH:/usr/local/git/bin" >> /etc/bashrc

> source /etc/bashrc

# Step 3

## Criação do Docker-Compose-Services