

## Embedando o Oracle Analytics Cloud em outros websites

O objetivo desse material é descrever de forma simples como é possível acessar uma visualização do OAC diretamente através de outros portais, inclusive utilizando mecanismos para que o processo de login não precise ser feito manualmente.

Existem 2 maneiras de embedar o conteúdo de uma visualização ou dashboard em uma página: Utilizando iFrame ou o Framework Javascript. Ambos são descritos na documentação abaixo:

<https://docs.oracle.com/en/cloud/paas/analytics-cloud/acubi/embed-analytics-content-applications-and-web-pages.html>

Via iFrame basta copiar a URL da página contendo o projeto e colocá-la na página utilizando iFrame. É um processo bastante trivial.

Via Javascript o processo é um pouco mais longo e passa pelas etapas:

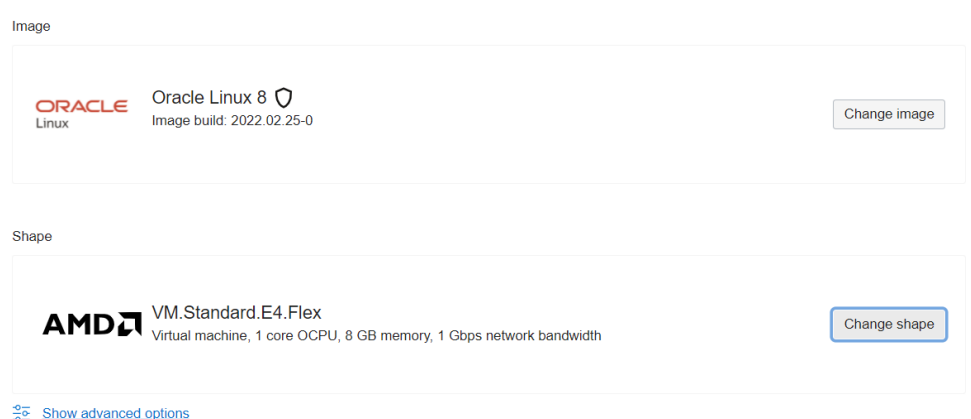
- Criar uma página HTML para receber a visualização
- Adicionar o endereço IP da origem como Safe Domain no OAC
- Preparar o Embedding na página HTML
- Determinar o modo de autenticação

### Criando uma página HTML para receber a visualização

Caso você já possua uma página pronta pode passar diretamente para a parte do código HTML. Como não a possuímos, criaremos uma máquina virtual para hospedá-la:

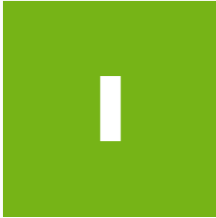
A instância não tem pre-requisitos. No exemplo utilizarei Oracle Linux 8 em uma shape AMD.

Não se esqueça de habilitar um endereço IP Público para a máquina.



Após o provisionamento, faça login via ssh no endereço IP da máquina, que pode ser encontrado na descrição da mesma no momento em que a instância fica disponível.

Compute » Instances » Instance details » Work requests



RUNNING

### EmbeddingOAC

Start Stop Reboot Edit More Actions

Instance information Shielded instance Oracle Cloud Agent Tags

#### General information

Availability domain: AD-1  
Fault domain: FD-3  
Region: sa-saopaulo-1  
OCID: ...fzss4a Show Copy

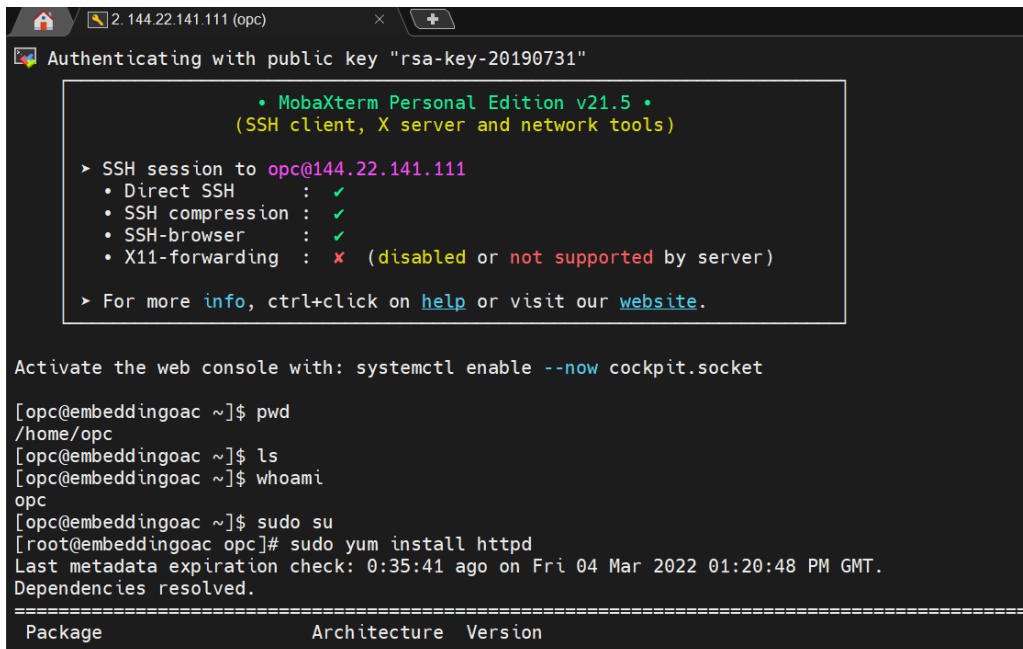
#### Instance access

You connect to a running Linux instance using a Secure Shell (SSH) connection. You'll private key from the SSH key pair that was used to create the instance.

Public IP address: 144.22.141.111 Copy  
Username: opc

Acesse a máquina via ssh e instale o Apache Server para subir sua página. Utilize o código abaixo:

```
sudo yum install httpd
sudo systemctl enable httpd
sudo systemctl restart httpd
sudo firewall-cmd --permanent --add-port=80/tcp
sudo firewall-cmd --reload
```



```
2. 144.22.141.111 (opc)
Authenticating with public key "rsa-key-20190731"
• MobaXterm Personal Edition v21.5 •
  (SSH client, X server and network tools)
> SSH session to opc@144.22.141.111
  • Direct SSH : ✓
  • SSH compression : ✓
  • SSH-browser : ✓
  • X11-forwarding : ✗ (disabled or not supported by server)
> For more info, ctrl+click on help or visit our website.

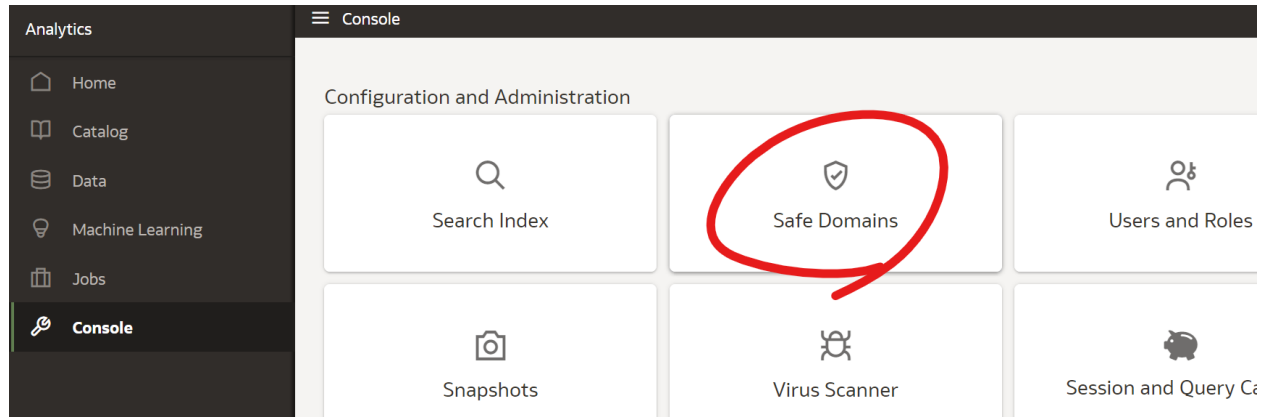
Activate the web console with: systemctl enable --now cockpit.socket

[opc@embeddingocac ~]$ pwd
/home/opc
[opc@embeddingocac ~]$ ls
[opc@embeddingocac ~]$ whoami
opc
[opc@embeddingocac ~]$ sudo su
[root@embeddingocac opc]# sudo yum install httpd
Last metadata expiration check: 0:35:41 ago on Fri 04 Mar 2022 01:20:48 PM GMT.
Dependencies resolved.
=====
Package                Architecture Version
=====
```

Uma vez instalado, pode-se acessar o endereço de IP Público via browser para avaliar se de fato esse passo foi bem sucedido. Não se esqueça de liberar as portas 80 e 443 na VCN associada à instância.

## Adicionar o endereço IP da origem como Safe Domain no OAC

Dentro do seu ambiente OAC, vá até Menu > Console > Safe Domains



Inclua o endereço de IP Público da sua instância e o habilite para todas as funções.

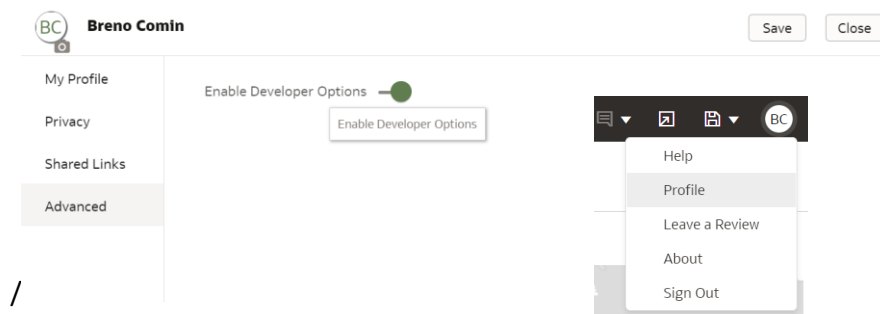
A screenshot of the 'Safe Domains' configuration page. It features a table with columns: Domain Name, Image, Allow Frames, Script, Font, Style, Media, Connect, Embedding, and Delete. The first row is 'All domains' with all checkboxes checked. The second row shows a specific domain '144.22.141.111' with all checkboxes also checked. Below the table is an 'add domain' button.

Domain Name	Image	Allow Frames	Script	Font	Style	Media	Connect	Embedding	Delete
All domains	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
144.22.141.111	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<a href="#">add domain</a>									

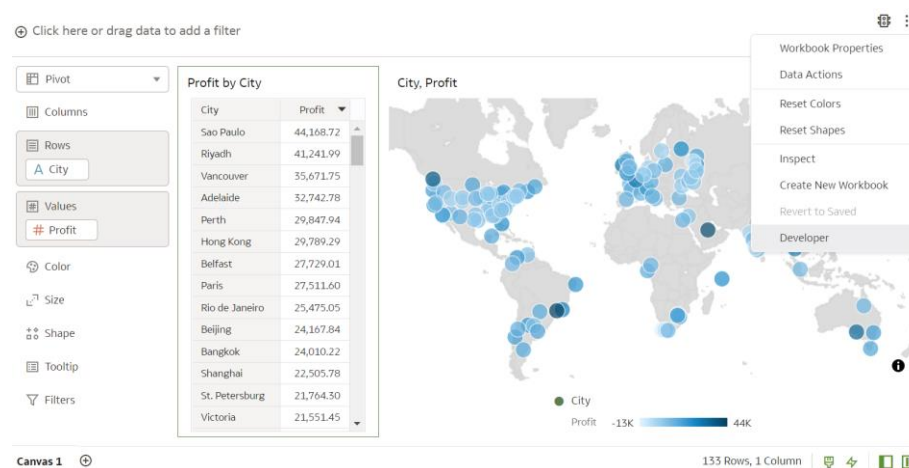
## Preparar o Embedding na página HTML

Agora podemos atualizar nossa página para que ela receba o código HTML. Primeiro, certifique-se de que as opções de Desenvolvedor estão habilitadas no seu OAC. Isso pode ser feito clicando no superior esquerdo da tela (em qualquer página e selecionando a opção Profile)

Dentro das opções selecione Advanced e habilite as opções de Developer.



Feito isso, acesse qualquer visualização ou dashboard, clique nos três pontos na região superior esquerda da tela e selecione a opção Developer.



Essa opção abrirá o espaço abaixo, e ao selecionar a aba Embed, teremos a informação necessária para realizar tal tarefa. É importante notar que esse não é o código todo que será utilizado na página HTML.

Performance Tools JSON XML Embed Datasets Dataset UI Options

#### Scripts to Include

Include the following <script> tag once per page. Based on the JavaScript framework used, replace <embeddingMode> with either "jet" or "standalone"

```
<script src="https://oracleanalyticsinstancejh-grapsblwflf-gr.analytics.ocp.oraclecloud.com/public/dv/v1/embedding/<embeddingMode>/embedding.js" type="application/javascript"></script>
```

Copy

For the 'standalone' embedding use case, include the following <script> tag to apply Knockout bindings after project is fully loaded. This should be executed in a body onLoad handler or after the <oracle-dv> tag. If embedding into an Oracle JET application, please refer to the product documentation for details.

```
<script>
  requirejs(['knockout', 'ojs/ojcore', 'ojs/ojknockout', 'ojs/ojcomposite', 'jet-composites/oracle-dv/loader'],function(ko) {
    ko.applyBindings();
  });
</script>
```

Copy

#### Default

To embed this project showing the default view, use the following HTML inside an appropriately sized element:

```
<oracle-dv project-path="/Catalog/users/breno_rc@hotmail.com/Dashboard Vendas - Branco">
</oracle-dv>
```

Copy

Para construir o código, gosto muito de recorrer à documentação do OAC onde há um exemplo pronto abordando a prática de embedar o HTML em uma página genérica:

<https://docs.oracle.com/en/cloud/paas/analytics-cloud/acubi/embed-javascript.html#GUID-FFBB4351-F80B-453A-9CC3-E3AA4B42539F>

```

<!DOCTYPE html>
<html dir="ltr">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>Embedded Oracle Analytics Project Example</title>
    <script src="https://<instance>.analytics.ocp.oraclecloud.com/public/dv/v1/embedding/<embedding
mode>/embedding.js" type="application/javascript">
    </script>

  </head>
  <body>
    <h1>Embedded Oracle Analytics Project</h1>
    <div style="border:1px solid black;position: absolute; width: calc(100% - 40px); height: calc(100% -
120px)" >
      <!-- The following tag is the tag that will embed the specified project.
Verify the project-path is the same as the server you are hosting this project on. -->
      <oracle-dv
        project-path="<project path>"
        active-page="canvas"
        active-tab-id="1">
      </oracle-dv>
    </div>
    <!--Apply Knockout bindings after DV project is fully loaded. This should be executed in a body onload
handler or in a <script> tag after the <oracle-dv> tag.
-->
    <script>
      requirejs(['knockout', 'ojs/ojcore', 'ojs/ojknockout', 'ojs/ojcomposite', 'jet-composites/oracle-
dv/loader'], function(ko) {
        ko.applyBindings();
      });
    </script>
  </body>
</html>

```

Podemos utilizar esse arquivo HTML, desde três mudanças sejam realizadas nos segmentos em destaque:

1. Alterar o nome da instância onde o código traz o valor <instance>
2. Definir o modo de embedding utilizado. Há apenas duas opções possíveis: /jet/ para embedar em aplicações Oracle JET ou /standalone/ para aplicações que não utilizam Oracle JET (maioria dos casos).
3. Atualizar o Project Path para seu Workbook. Isso pode ser retirado diretamente da área do Developer.

Abaixo um exemplo dos dois trechos atualizados:

```

<script src="https://workshop-grri30nzv1ul-
gr.analytics.ocp.oraclecloud.com/public/dv/v1/embedding/standalone/embedding.js"
type="application/javascript">    </script>

```

```

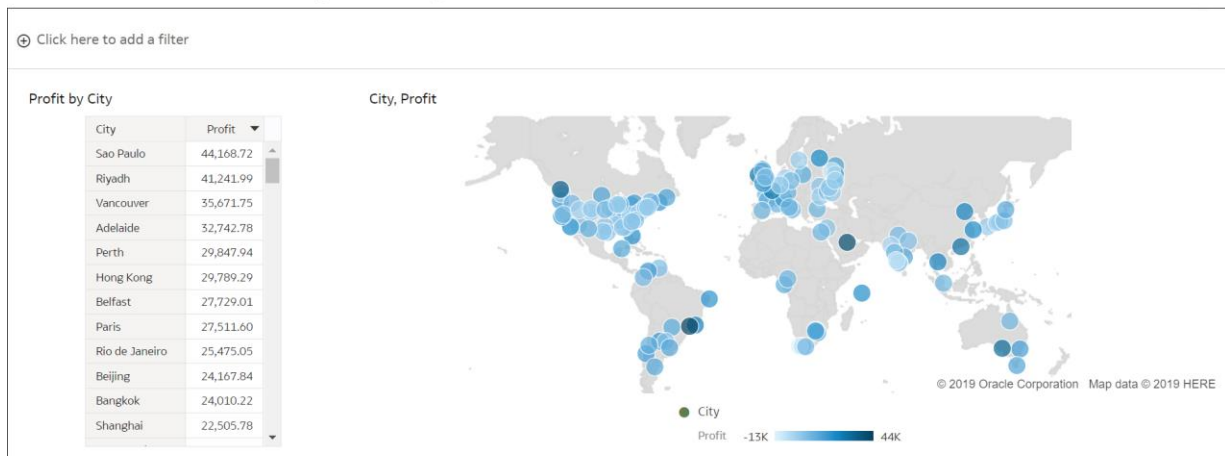
<oracle-dv    project-path="@Catalog/users/breno.comin@oracle.com/Sample Project"
              active-page="canvas"
              active-tab-id="1">          </oracle-dv>

```

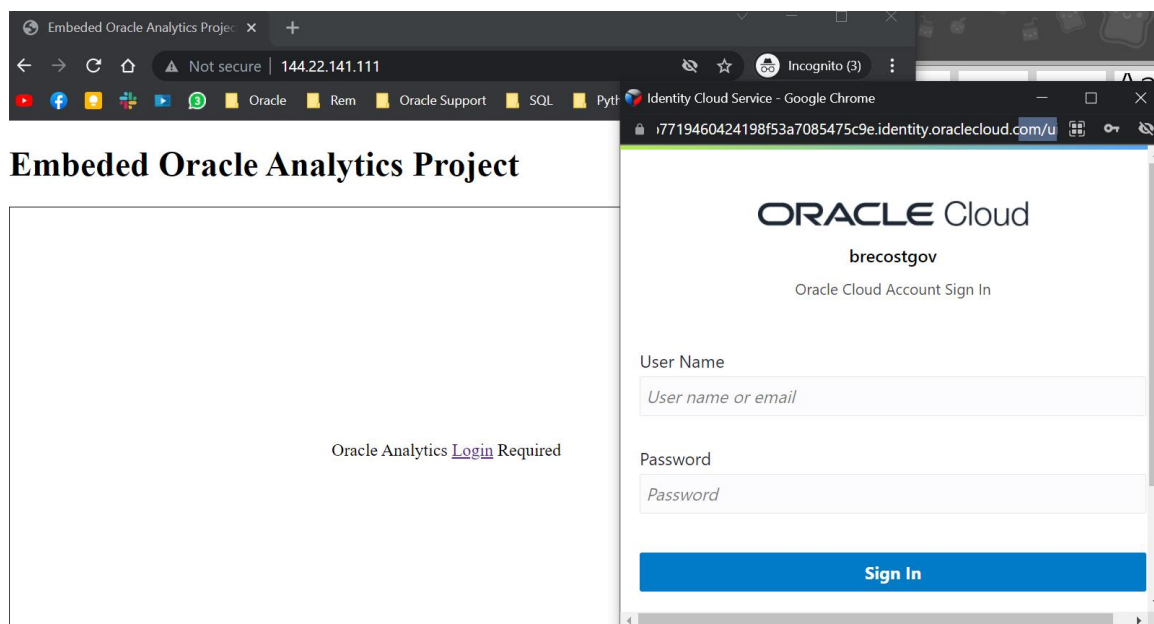
Salve o novo código como index.html e o copie para a máquina virtual criada anteriormente.

Acessar o endereço vai te trazer para uma página com o OAC embedado, na qual você pode interagir com as visualizações criadas no seu ambiente

## Embeded Oracle Analytics Project



Note que caso você não tenha feito login anteriormente, suas credenciais serão solicitadas no momento em que você tentar se conectar com o ambiente.



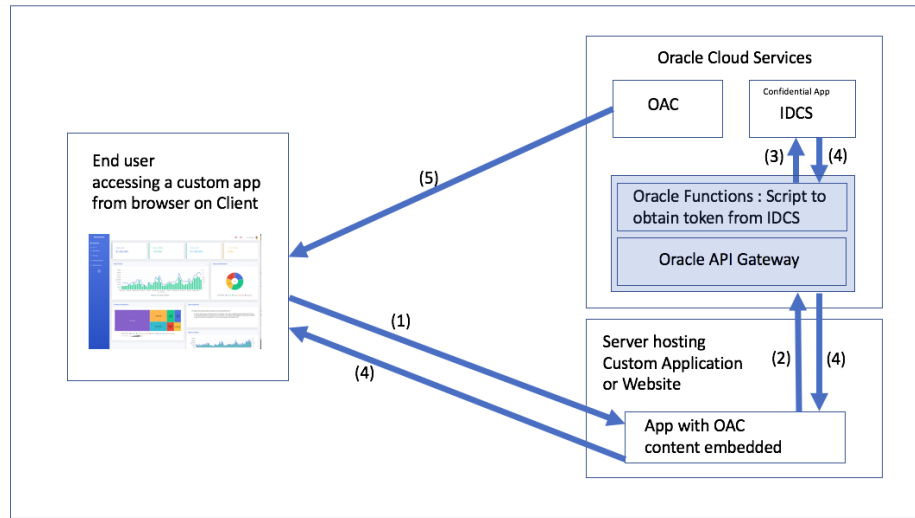
No próximo passo veremos como automatizar esse login para que não seja necessário colocar o usuário e senha manualmente todas as vezes que acessarmos a página.

## Automatização de login no Oracle Analytics Cloud

(baseado [nesse artigo](#))

Os acessos ao Oracle Analytics Cloud são gerenciados pelo IDCS (Oracle Identity Cloud Service). O login pode ser feito incluindo as credenciais de username e password, ou então pela geração de um token, que valida o acesso. O processo detalhado abaixo vai utilizar esse método.

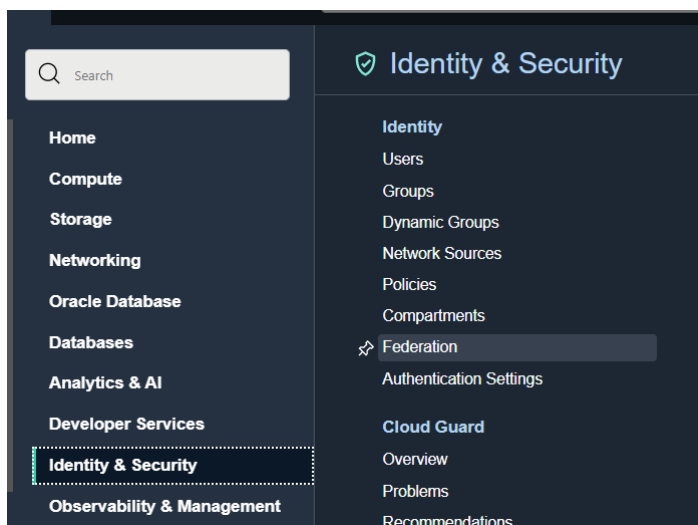
Para chamá-lo a partir de um host que não necessariamente estará na OCI, utilizaremos o API Gateway que ativa uma Function para gerar o token.



## Configurando o IDCS

O primeiro passo é configurar o IDCS para geração do token supracitado.

Acesse **Federation** dentro da área **Identity & Security**



Clique em OracleIdentityCloudService e no link para o serviço de IDCS.

Identity » Federation » Identity Provider Details

## OracleIdentityCloudService

[Edit Provider Details](#) [Reset Credentials](#) [Add Tags](#) [Delete](#)

**Identity Provider Information** **Tags**

**OCID:** ...2b4jqa [Show](#) [Copy](#) **Description:** Oracle identity cloud service added during account crea

**Created:** Thu, Feb 24, 2022, 12:17:51 UTC **Type:** IDCS

**Encrypt Assertion:** Disabled **Force Authentication:** Disabled

**Oracle Identity Cloud Service Console:** <https://idcs-2a18b7719460424198f53a7085475c9e.identity.oraclecloud.com/ui/v1/adminconsole>

**IDCS service identifier:** 0f2260412c... [Show](#) [Copy](#)

**Authentication Contexts:** -

Dentro do IDCS, clique no **Menu Hamburguer** (superior esquerdo) e em **Applications**:

**Dashboard**

- Users
- Groups
- Applications**
- Oracle Cloud Services
- Jobs

**ORACLE** Identity Cloud Serv

Welcome breno\_rc@hotmail

[Watch the Video](#) [Learn More](#)

[What's New](#)

Filter by Date Range [Last 30 Days](#) ▼



Adicione então uma **Aplicação Confidencial** e a configure da seguinte maneira:

- Habilite **Resource Owner** em **Allowed Grant Types**

ORACLE Identity Cloud Service

### Add Confidential Application

Back Details Client Resources Authorization Next

Configure this application as a client now Skip for later

#### Authorization

Allowed Grant Types ☒ Resource Owner ☐ Client Credentials ☐ JWT Assertion ☐ SAML2 Assertion ☐ Refresh Token ☐ Authorization Code

☐ Implicit ☐ Device Code

Allow non-HTTPS URLs ☐

Redirect URL

Logout URL

Post Logout Redirect URL

Security ☐ Trusted Client ☐ Certificate

Allowed Operations ☐ Introspect ☐ On behalf Of

Bypass Consent ☐

#### Token Issuance Policy

Authorized Resources ☐ All ☐ Tagged ☒ Specific

- Desça até a área de **Resources**, clique em **Add Scope** e selecione a sua instância de OAC

Token Issuance Policy

Authorized Resources ☐ All ☐ Tagged ☒ Specific

Resources

Resource

ANALYTICSINST\_orac

Grant the client access

Select All

<input checked="" type="checkbox"/>		ANALYTICSINST_oac22-grapsblwwfif-gr	<input type="button" value="&gt;"/>
<input type="checkbox"/>		ANALYTICSINST_oracleanalyticsinstancejh-grapsblwwfif-gr	<input type="button" value="&gt;"/>
<input type="checkbox"/>		ANDC	<input type="button" value="&gt;"/>
<input type="checkbox"/>		AUTOANALYTICS	<input type="button" value="&gt;"/>
<input type="checkbox"/>		BotSaaSAuto	<input type="button" value="&gt;"/>

Page 1 of 1 (1-14 of 14 items) < 1 >

- Vá até a **última aba** e clique em **Finish**

## Add Confidential Application

< Back

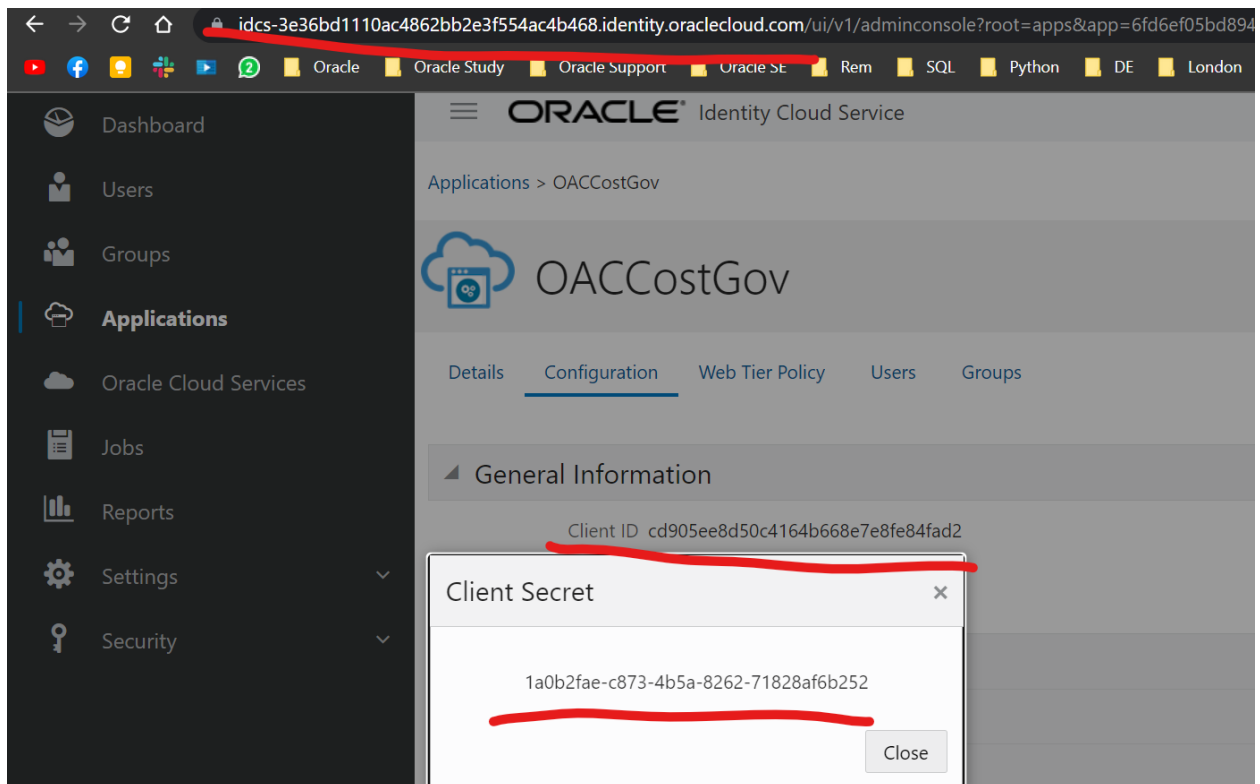
Details Client Resources **4** Authorization

Enforce Grants as Authorization ☐

Agora podemos testar a geração do token. Isso pode ser feita via Postman. Ele nos ajudará também nos próximos passos. Uma solicitação cURL também pode ser utilizada.

De uma forma ou de outra, os valores necessários são os mesmos. Garanta que você os possui antes de prosseguir:

- URL do seu serviço de IDCS
- Client ID e Client Secret da sua aplicação



- Scope da Aplicação no IDCS

## Token Issuance Policy ⓘ

Authorized ☐ All  
 Resources ☐ Tagged  
☒ Specific

### Resources

+ Add Scope

	Protected	Scope
ohssmb-ia	No	<a href="https://eza73dwhd2khiccm4vavtfrm5vffe7sa.analytics.ocp.oraclecloud.comurn:opc:resource:">https://eza73dwhd2khiccm4vavtfrm5vffe7sa.analytics.ocp.oraclecloud.comurn:opc:resource:</a>

- Username e password com acesso ao OAC

## Validação via Postman:

- Crie uma request do tipo POST e inclua as informações de acordo com as screenshots:

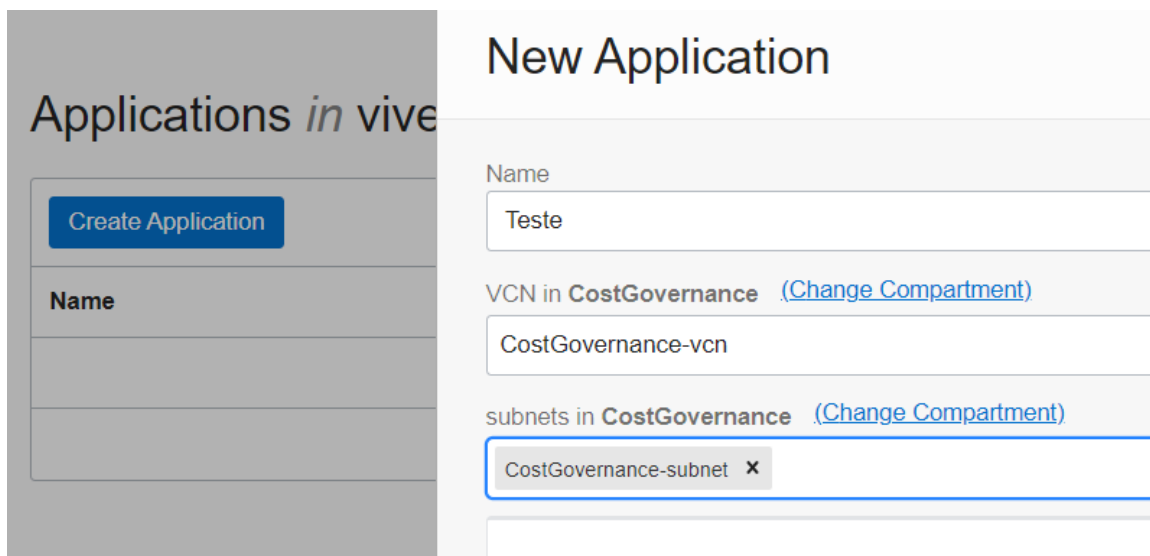
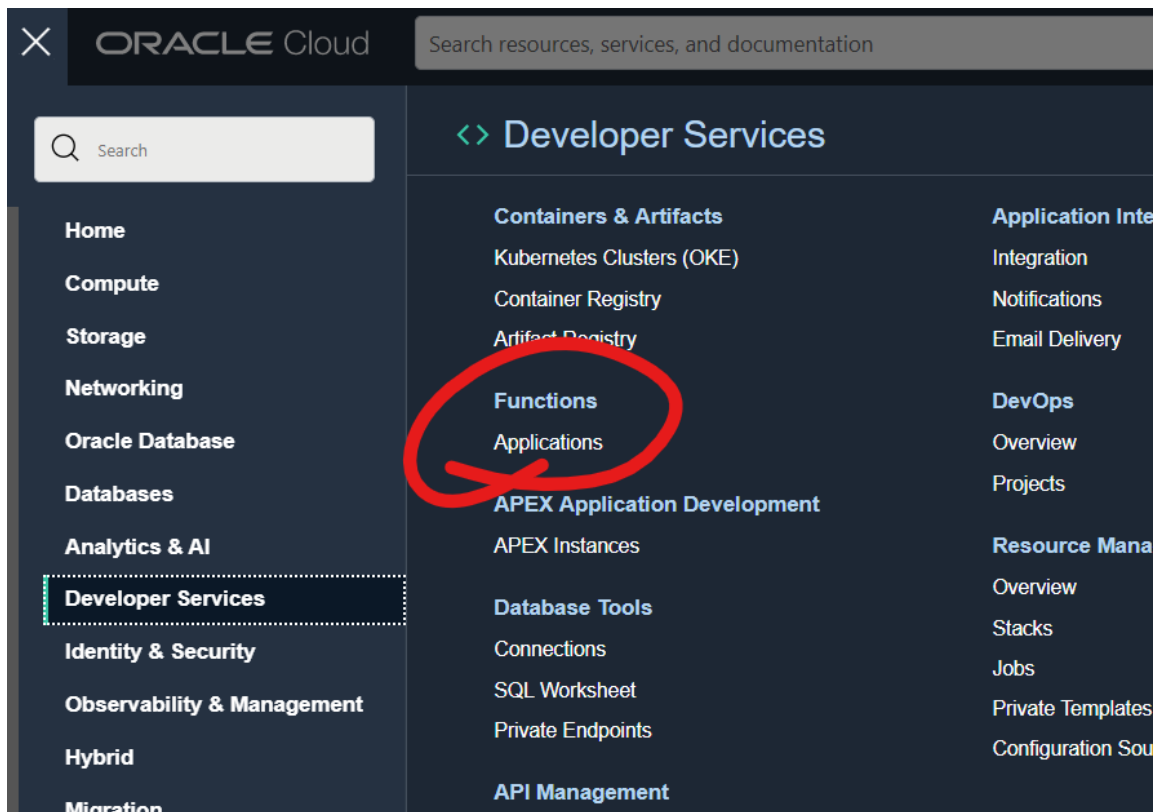
The screenshot shows a Postman interface for a POST request to the IDCS endpoint: `https://idcs-2a18b7719460424198f53a7085475c9e.identity.oraclecloud.com/oauth2/v1/token`. The request is configured with Basic Authentication. The Username field contains the Client ID `374cc1492e7c4d30bdd24268c6a6dfac`, and the Password field contains the Client Secret `cac7f63f-5489-48bc-be13-0b5fb3811ef0`. A warning message states: "Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables". A "Show Password" checkbox is checked.



## Utilizando OCI Functions para gerar o seu Token

Primeiramente crie uma nova aplicação do Functions no seu ambiente. Utilize uma VCN padrão e um nome qualquer.

Clique em **Developer Services** -> **Functions** e depois em **Create Application**.

A screenshot of the 'New Application' form in the Oracle Cloud console. The form is titled 'New Application' and has a 'Create Application' button. It contains several input fields: 'Name' with the value 'Teste', 'VCN in CostGovernance' with the value 'CostGovernance-vcn' and a link to '(Change Compartment)', and 'subnets in CostGovernance' with the value 'CostGovernance-subnet' and a link to '(Change Compartment)'. The 'subnets' field has a small 'x' icon next to the value, indicating it can be removed or edited. The form is partially obscured by a sidebar on the left with the title 'Applications in vivo'.

Realize o passo-a-passo do **Getting Started** até o passo de número 7:

### Getting started

Cloud Shell setup  
Quickly create, deploy and invoke functions using Cloud Shell

Local setup  
Set up a development machine to create, deploy and invoke functions

Begin your Cloud Shell session

[Learn more about Cloud Shell](#)

1 Launch Cloud Shell

Setup fn CLI on Cloud Shell

2 Use the context for your region

```
fn list context
fn use context us-ashburn-1
```

Copy

Copy

3 Update the context with the function's compartment ID

```
fn update context oracle.compartment-id ocid1.tenancy.oc1..aaaaaaaajckr6smyb5ofkptbhbunqvhcqbqvjdazubyhfkdfoklhxwa
```

Copy

4 Provide a unique repository name prefix to distinguish your function images from other people's. For example, with 'jdoe' as the prefix, the image path for a 'hello' function image is '<region-key>.ocir.io/<tenancy-namespace>/jdoe/hello:0.0.1'

```
fn update context register fnf ocir.io/tenancy-namespace/jdoe
```

Copy

Chegando no passo 8 você não tem a necessidade de realizá-lo, pois trata-se de um exemplo 'Hello World'. Vamos diretamente para nosso código.

Execute então os comandos:

- Criação de um diretório

```
mkdir oactokengen
cd oactokengen/
```

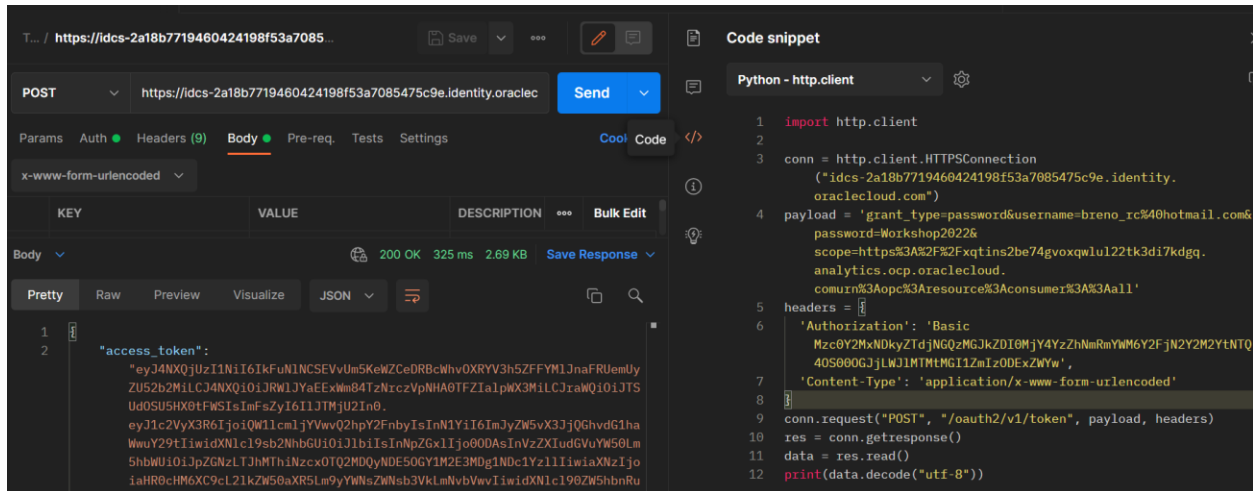
- Criação de uma função em Python (se preferir pode usar outra linguagem de programação, mas terá que utilizar seu próprio código).

```
fn init --name oactokengen --runtime python
```

- Edite o arquivo .py

```
vim func.py
```

- Inclua o comando em python que irá gerar o token (O Postman gera esse comando automaticamente, basta clicar no botão **Code** no lado direito e selecionar a linguagem na qual deseja fazer essa chamada)



No caso do Python, o código deverá ser indentado e adequado ao padrão que já vem estabelecido para a função. Veja o exemplo abaixo:

```
import io
import json
import logging

from fdk import response

def handler(ctx, data: io.BytesIO = None):
    import http.client

    conn = http.client.HTTPSConnection("ids-2a18b7719460424198f53a7085475c9e.identity.oraclecloud.com")
    payload = 'grant_type=password&username=breno_rc%40hotmail.com&password=Workshop2022&scope=https%3A%2F%2Ffxqtins2be74gvoxqwlul22tk3di7kdqg.analytics.ocp.oraclecloud.comurn%3Aopc%3Aresource%3Aconsumer%3Aaall'
    headers = {
        'Authorization': 'Basic Mzc0Y2MxNDkyZTdjNGQzMGI0MjY4YzZhNmRmYWM6Y2FjN2Y2M2YtNTQ4OS00OGJjLWJlMTMTMG1lZmIzODExZWYw',
        'Content-Type': 'application/x-www-form-urlencoded'
    }

    conn.request("POST", "/oauth2/v1/token", payload, headers)
    res = conn.getresponse()
    data = res.read()
    return(data.decode("utf-8"))
```

Para terminar a edição no vim, basta clicar em **ESC**, seguido de **:wq** e **ENTER**

Faça então o deploy da sua aplicação (**Utilize o nome da Aplicação e não da Função !!!**)

```
fn -v deploy --app EmbedOAC
```

```
Parts: [gru.ocir.io/grapsblwflf/breno/oactokengen:0.0.2]
Using Container engine docker to push
Pushing gru.ocir.io/grapsblwflf/breno/oactokengen:0.0.2 to docker registry...The push refers to repository [gru.ocir.io/grapsblwflf/breno/oactokengen]
3e4bb9590003: Pushed
9832fe33ee76: Pushed
d353a11c2755: Pushed
9841f39dc3ea: Pushed
63fee7088442: Pushed
5d83175b427c: Pushed
e57f007acf74: Pushed
0.0.2: digest: sha256:ef13a7869005adf8093bb2de35e129f271440e5c66494c25dcbe09e1284e3ba5 size: 1780
Updating function oactokengen using image gru.ocir.io/grapsblwflf/breno/oactokengen:0.0.2...
Successfully created function: oactokengen with gru.ocir.io/grapsblwflf/breno/oactokengen:0.0.2
brno_rc@cloudshell:oactokengen (sa-saopaulo-1)$
```

O modelo da chamada é fn invoke <nome-da-aplicação> <nome-da-função>

O resultado poderá ser visto na própria console:

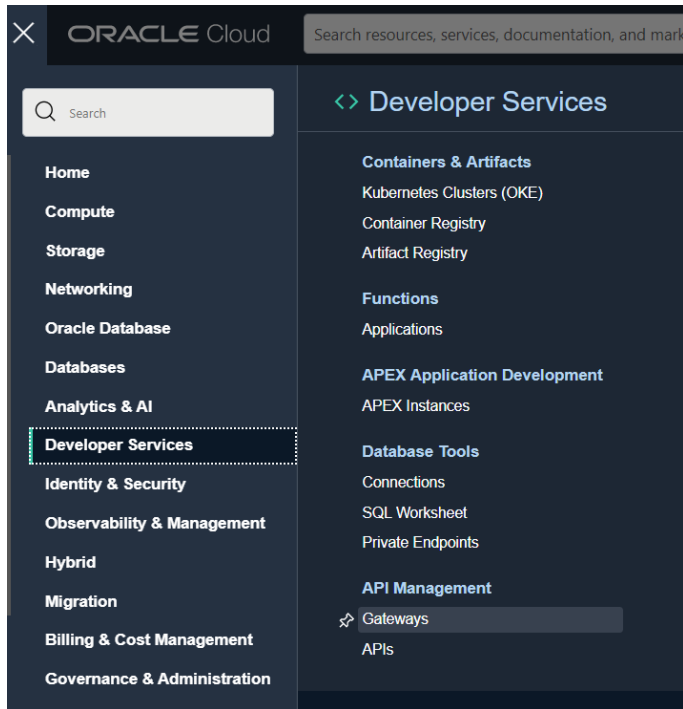
[illegible]

Temos agora um método confiável para gerar nosso token em tempo real ao acessar nossas páginas. Agora só falta fazer nossa página em que o OAC está embeddado chamar essa função. Para viabilizar esse processo, criaremos um API Gateway.



## Criando e Configurando o API Gateway

Vá para **Developer Services -> API Management -> Gateways**



**Crie seu Gateway em uma Rede Pública:**

The screenshot shows the 'Create Gateway' form. On the left is a sidebar with the title 'Gateways' and a 'Create Gateway' button. Below it is a table with columns 'Name' and 'OAC\_Validation'. The main form has the following fields: NAME (EmbeddingOAC), TYPE (Public), COMPARTMENT (brecostrgov (root)), and a 'Network' section. The 'Network' section includes a description 'Configure how the API Gateway attaches to your network.' with a link 'Learn more about Networking'. It also has two dropdown menus: 'VIRTUAL CLOUD NETWORK IN BRECOSTRGOV (ROOT) (CHANGE COMPARTMENT)' with 'FunctionsNetwork' selected, and 'SUBNET IN BRECOSTRGOV (ROOT) (CHANGE COMPARTMENT)' with 'Public Subnet For Oracle Functions' selected. At the bottom are 'Create Gateway' and 'Cancel' buttons.

Quando o Gateway estiver disponível, vá até Deployment e **Crie um novo Deployment**

Dê um nome e um path prefix para seu Deployment:

Basic Information	
NAME	<input type="text" value="EmbeddingOAC"/>
PATH PREFIX ⓘ	<input type="text" value="/embedoac"/>
COMPARTMENT	<input type="text" value="brecoastgov (root)"/>

Adicione o endereço que receberá a página com o Analytics nas políticas de CORS:

CORS Policy	
Origins	
ALLOWED ORIGINS ⓘ	<input type="text" value="https://144.22.141.111/"/>
	<input type="text" value="http://144.22.141.111/"/>
Methods	
ALLOWED METHODS <small>OPTIONAL</small> ⓘ	<input type="text" value="GET x"/>

Route 1

PATH ⓘ

/idcstoken

METHODS

ANY x

TYPE

Oracle Functions

Specifies the type of the backend service. [Learn more](#) about the Oracle Functions backend.

APPLICATION IN BRECO\$TGOV (ROOT) [\(CHANGE COMPARTMENT\)](#)

EmbedOAC

FUNCTION NAME

oactokengen

Quando o Deployment ficar verde é sinal que ele foi criado com sucesso.

## Deployments

Create Deployment

Name	Path Prefix	State	Endpoint	Deployed
<a href="#">EmbeddingOAC</a>	/embedoac	● Active	...m/embedoac <a href="#">Show</a> <a href="#">Copy</a>	Wed, Apr 13, 2022, 21:06:56 UTC

Showing 1 Item < 1 of 1 >

API EmbeddingOAC is being deployed

API EmbeddingOAC was successfully deployed

[illegible]

Se a chamada não funcionar, revise o API Gateway e as regras de rede definidas para a rede virtual em que ele está rodando. Habilitar a porta 443 é importante para que a sua solicitação não seja bloqueada pela rede.

## Atualizando a página web

O último passo é agora atualizar a página web para que ao invés da validação padrão (via usuário), ela chame o API Gateway e utilize o token gerado para validar o acesso ao OAC.

O passo-a-passo para hospedar uma webpage em uma máquina virtual e preparar o servidor já foi descrito nesse arquivo, portanto vamos diretamente ao novo código HTML.

Perceba que esse arquivo possui algumas alterações gráficas realizadas com a tag `<style>`. Elas não são de forma alguma necessárias. Apenas decidi mantê-las a título de curiosidade.

Muito erros podem acontecer ao se tentar realizar o Embedding do OAC em outras páginas web, mas na minha experiência os maiores motivos para eles são:

- Ausência de políticas CORS configuradas no OAC (Safe Domains)
- Ausência de políticas CORS configuradas no API Gateway.
- Problemas na rede em que o API Gateway foi criado
- Erros pontuais no arquivo HTML

Certifique-se de revisar cada um desses pontos e muito provavelmente essa configuração do Analytics embeddado em outra página ou aplicação funcionará 100%.

```

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Oracle Analytics Embed Example Using a Token</title>
<meta id="viewport" name="viewport" content="width=device-width, initial-scale=1, minimum-scale=1.0, maximum-
scale=1.0, user-scalable=no"/>
<style>
.bitech-layout-component {
background: black;
}
body
{
background-color: black !important;
}
h1 {
color: white !important;
margin-left: 40px !important;
}
.bi_filterbar_padding_left{
position: unset !important;
}
</style>
<script src="https://<instance>/public/dv/v1/embedding/standalone/embedding.js" type="application/javascript">
</script>
</head>

<body>
<h1>Sales Analytics</h1>
<div>
<oracle-dv
project-path="<your-project-path>"
active-page="canvas"
active-tab-id="1"
project-options='{ "bDisableMobileLayout":false, "bShowFilterBar":true}'>
</oracle-dv>
</div>

<script>
// Get bearer token from an API (e.g. Oracle API Gateway)
var Url = '<your-API-Gateway-URL>';
var token_request = new XMLHttpRequest();
token_request.open("GET", Url, false);
token_request.send(null);
{
if (token_request.status = 200) {
var idcs_token = JSON.parse(token_request.response).access_token;
requirejs(['jquery', 'knockout', 'obitech-application/application', 'ojs/ojcore', 'ojs/ojknockout',
'ojs/ojcomposite', 'jet-composites/oracle-dv/loader'],
function($, ko, application) {
application.setSecurityConfig("token", {tokenAuthFunction:
function(){
return idcs_token;
}
});
ko.applyBindings();
});
} else {
console.log('error ${token_request.status} ${token_request.statusText}')
}
}
</script>
</body>
</html>

```

## Apêndice – Validação da chamada API via comando cURL

Utilizar o comando abaixo:

```
curl --location --request POST 'https://<endereço-do-IDCS-até-o-.com>/oauth2/v1/token' \  
--header 'Authorization: Basic <ClientID:ClientSecret-codificados-em-base-64>' \  
--header 'Content-Type: application/x-www-form-urlencoded' \  
--data-urlencode 'grant_type=password' \  
--data-urlencode 'username=<seu-username-do-OAC>' \  
--data-urlencode 'password=<seu-password-do-OAC>' \  
--data-urlencode 'scope=<todo-o-scope-da-sua-aplicação-no-IDCS>'
```

Para codificar Client ID e Secret utilize o comando:

```
echo -n "<Client ID>:<Client Secret>" | base64
```

Exemplo pronto:

```
curl --location --request POST 'https://idcs-  
2a18b7719460424198f53a7085475c9e.identity.oraclecloud.com/oauth2/v1/token' \  
--  
header 'Authorization: Basic Mzc0Y2MxNDkyZTdjNGQzMGI0MjY4YzZhNmRmYWM6Y2FjN2Y2M2YtNTQ4OS  
00OGJjLWJlMTMtMGIlZmIzODExZWYw' \  
--header 'Content-Type: application/x-www-form-urlencoded' \  
--data-urlencode 'grant_type=password' \  
--data-urlencode 'username=meunome@hotmail.com' \  
--data-urlencode 'password=minhasenha' \  
--data-  
urlencode 'scope=https://xqtins2be74gvoxqwlul22tk3di7kdqg.analytics.ocp.oraclecloud.comurn:  
opc:resource:consumer::all'
```