

Módulo 3: Integração Salesforce

Neste laboratório, você usará o **Anypoint Studio** para criar uma aplicação para expor os dados do Salesforce. Para os propósitos deste laboratório, implementaremos a API do cliente representada pelos métodos GET e POST.

Etapas do laboratório

[Etapa 1: Criar um novo projeto Mule](#)

[Etapa 2: Executar o projeto API Customer](#)

[Etapa 3: Adicionar o conector Salesforce](#)

[Etapa 4: Criar uma operação de Query](#)

[Etapa 5: Conectar-se ao Salesforce](#)

[Etapa 6: Consultar os dados do objeto Account](#)

[Etapa 7: Testar a API do Salesforce](#)

[Etapa 8 \(Opcional\): Inserir um objeto no Salesforce](#)

[Etapa 9: Publicar a API no CloudHub](#)

[Etapa 10: Configurar arquivo de propriedades](#)

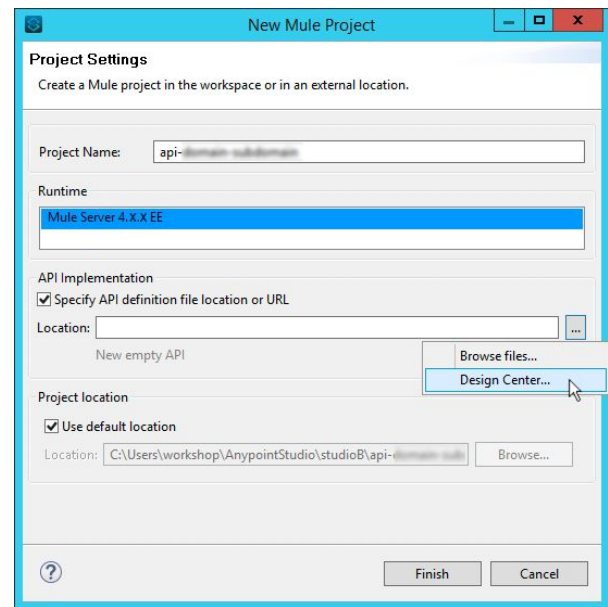
[Etapa 11: Proteger as credenciais do Salesforce](#)

Etapa 1: Criar um novo projeto Mule

Vamos usar nossa definição de API para criar a implementação. Para fazer isso, precisamos importar a especificação de RAML do **Design Center**.

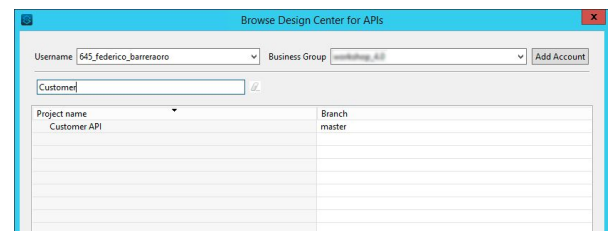
1. Inicie o Anypoint Studio no ícone da área de trabalho e selecione o mesmo espaço de trabalho em que você está trabalhando (exemplo: C:\workspaces\myworkspace).
2. No menu do Anypoint Studio, selecione **Arquivo > Novo > Projeto Mule** para criar um projeto. Uma janela será exibida para definir os detalhes desta nova aplicação.

3. Dê ao projeto o nome **api-customer**
4. Selecione o **Mule Server 4.xx EE**
5. Em API Implementation, selecione o checkbox e através do botão (...) selecione **Design Center**
6. Depois de clicar, realizar **login** com as mesmas credenciais utilizadas na plataforma Anypoint

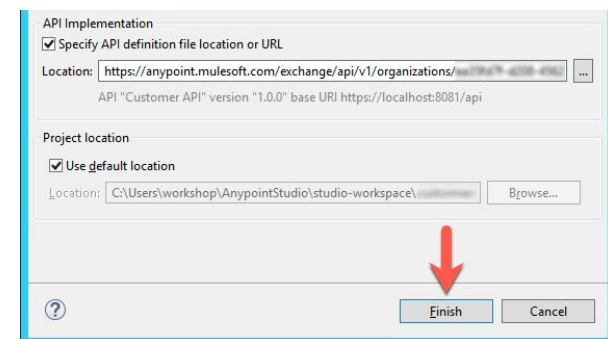


7. Pesquisar por **Customer API**

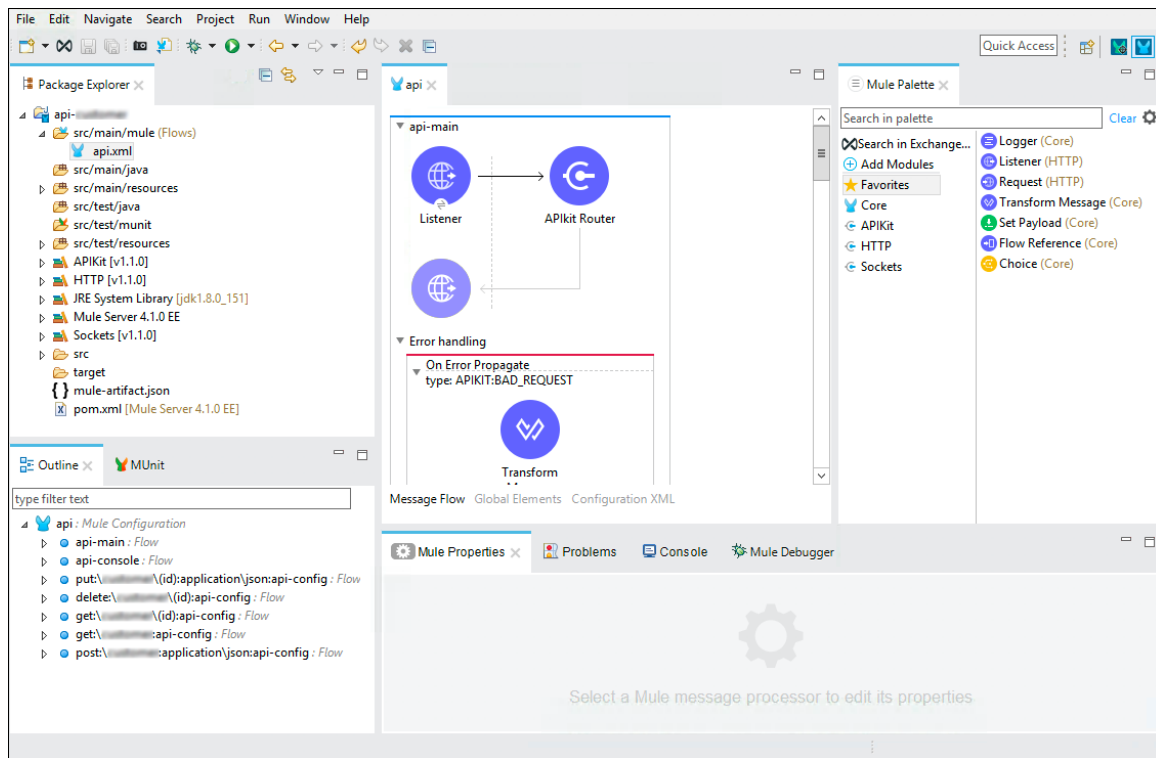
Nota: Se a API não for listada selecionar **Load more projects**.



8. Depois de selecionada a API finalizar a criação do projeto com o botão **Finish**



O Anypoint Studio cria seu novo projeto com a implementação gerada com base na sua especificação da API. A configuração gerada implementa o listener HTTP de entrada, bem como todos os recursos, metadados de tipo de dados de solicitação/resposta, etc.



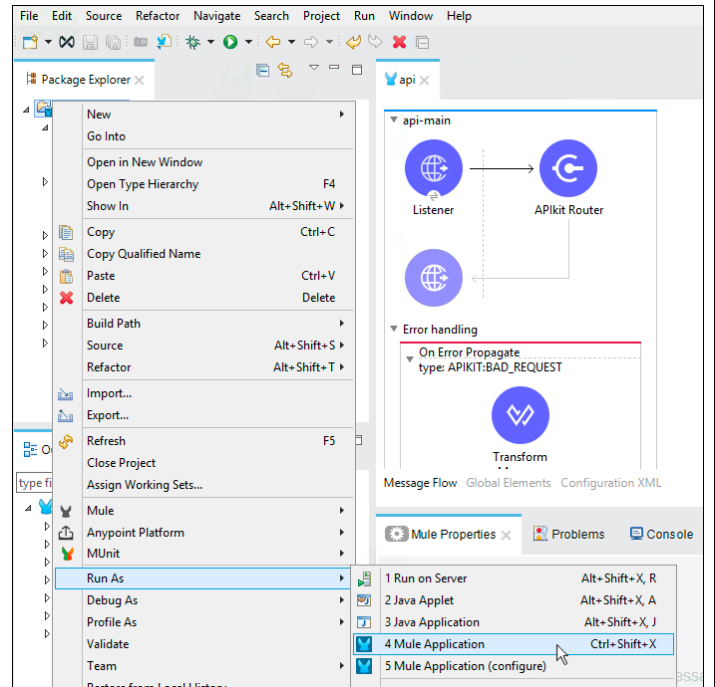
Esse projeto da **API Customer** está pronto para rodar!

Etapa 2: Executar o projeto API Customer

Vamos validar a API que acabou de ser criada na etapa anterior.

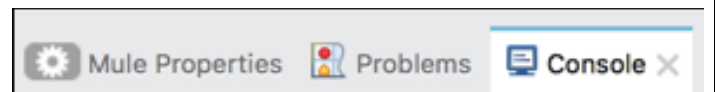
Para testar a API, vamos executá-la no **Anypoint Studio**.

1. No lado esquerdo, clique com o botão direito do mouse no projeto dentro de **Package Explorer**.
2. Selecione **Run As > Mule App**

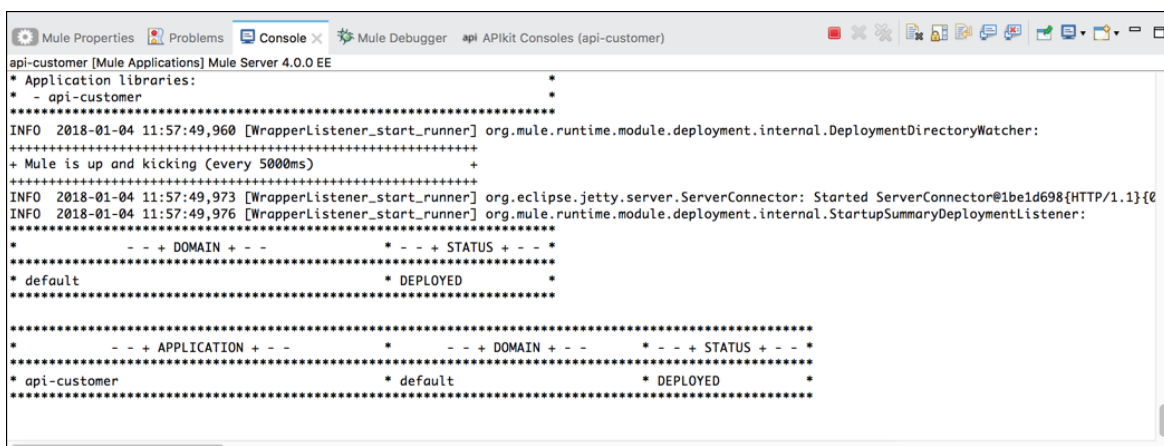


Toda vez que iniciarmos a aplicação, precisamos esperar que ele seja iniciada com sucesso.

3. Procure no painel central que possui **Mule Properties | Problems | Console**.



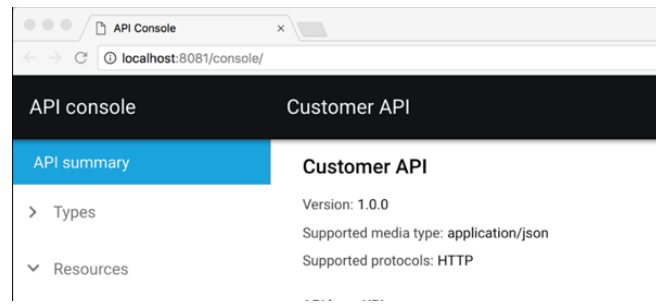
Selecione a guia **Console** para acompanhar o deployment. Após iniciar o runtime, a aplicação **api-customer** aparece como **"DEPLOYED"**.



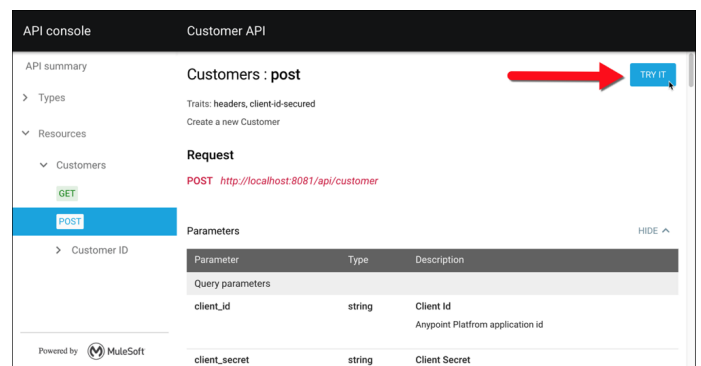
4. Uma vez que a aplicação está rodando, irá abrir a aba do **APIkit Console** no lado esquerdo
5. Clique em **Open console** para testar a aplicação



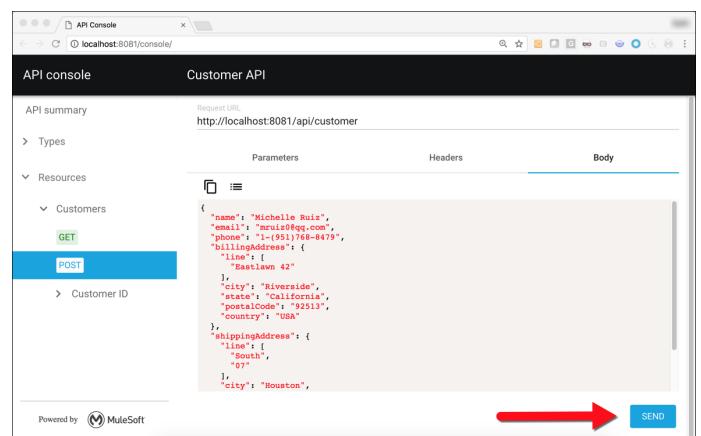
6. Uma janela do navegador abrirá com o endereço <http://localhost:8081/console/>



7. No lado esquerdo, clique no método **POST**. A documentação aparecerá no painel central
8. Em seguida, aperte o botão **Try It** no canto superior direito do painel central



9. Modifique o corpo da mensagem na aba **Body** ou mantenha o valor padrão
10. Clique em **Send**
11. Você receberá uma resposta da operação



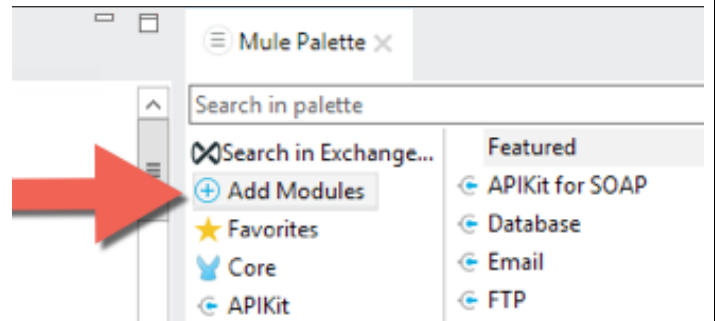
Ao final dessa etapa, vá para a aba **Console** e pressione o botão vermelho para parar o servidor.



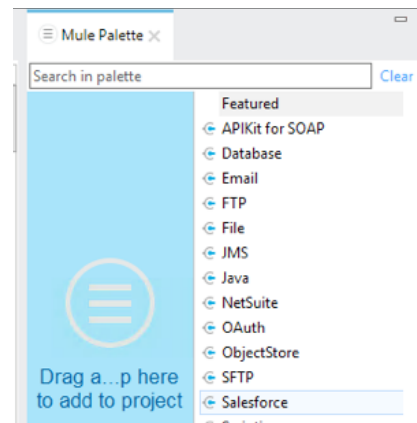
Etapa 3: Adicionar o conector Salesforce

A fim de agilizar o desenvolvimento, vamos adicionar o conector Salesforce disponível no **Anypoint Studio**.

1. Clique em **Add Module** para adicionar os módulos **Salesforce Connector** e **Validation** ao projeto.



2. Arraste e solte o **Salesforce Connector** da janela da direita para à esquerda



A tarefa de incluir novos conectores é simples. A partir de agora, podemos usar as operações Salesforce no projeto.

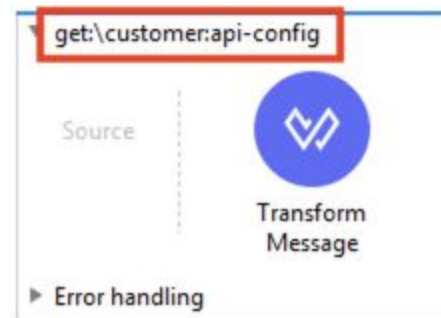
Etapa 4: Criar uma operação de Query

Agora você está pronto para implementar o método **GET** usando o conector do Salesforce.

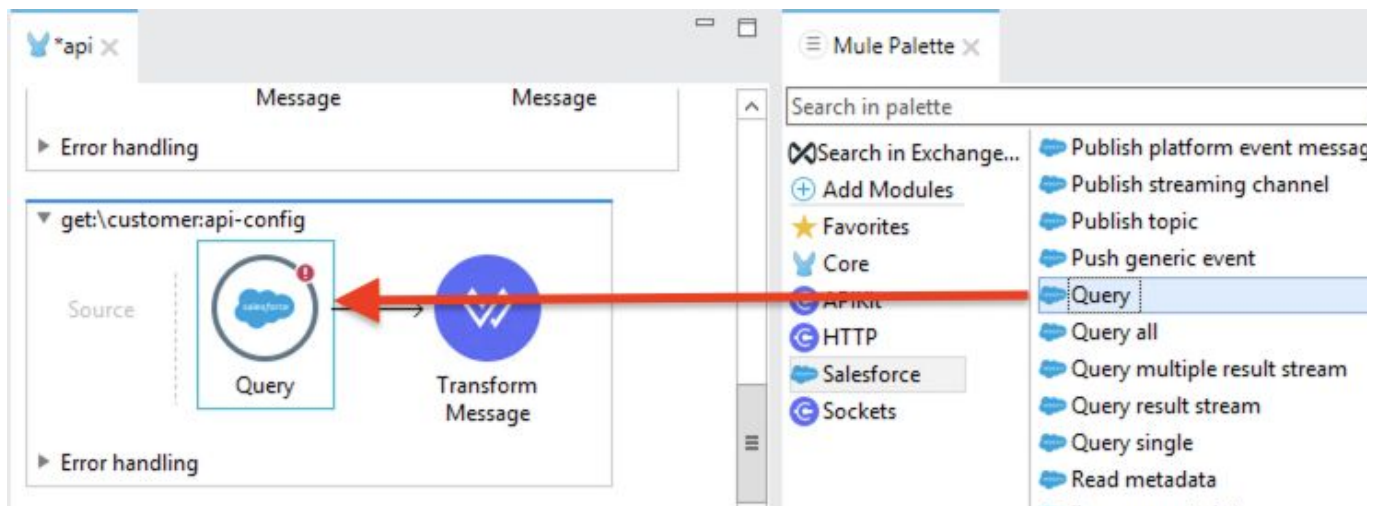
Identificamos o fluxo do projeto que corresponde ao método **GET /customer**.

1. Localize o fluxo **get:\customer:api-config**

Nesse fluxo, o operador representa a resposta estática para as requisições feitas à API. Vamos substituir por uma consulta dinâmica ao Salesforce.



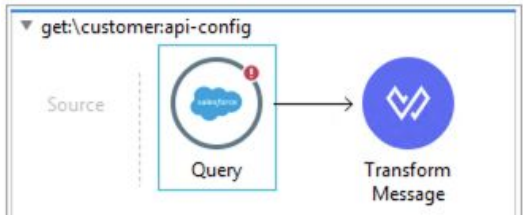
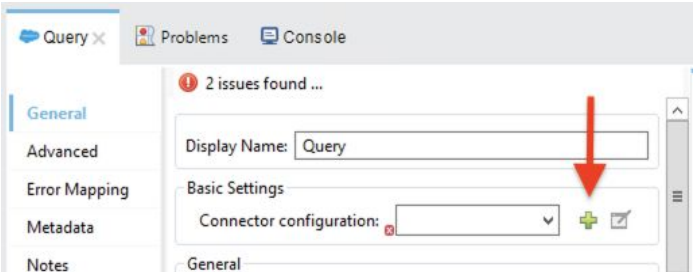

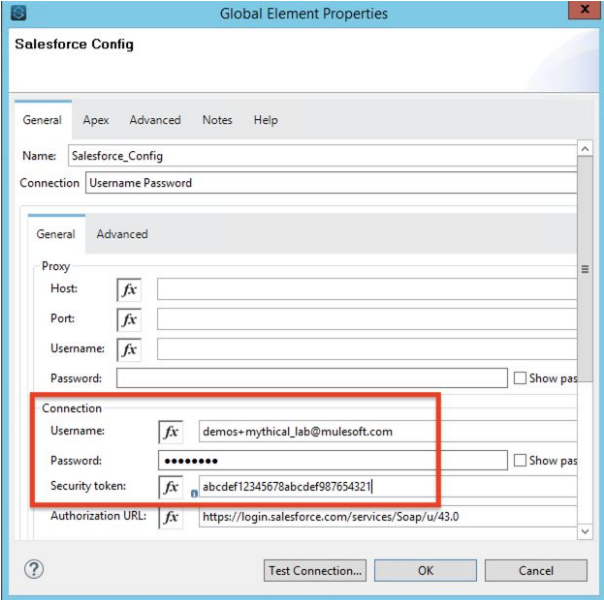
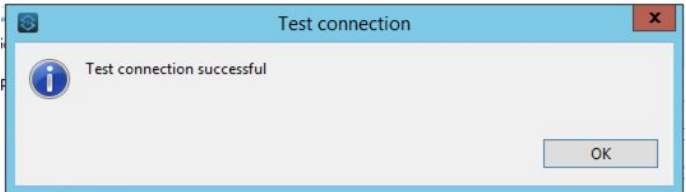
2. Em **Mule Palette**, selecione **Salesforce** e depois **Query**. Arraste e solte o componente para o início do fluxo.



A operação de consulta foi adicionada ao fluxo. Os próximos passos são configurar a conexão e escrever a consulta.

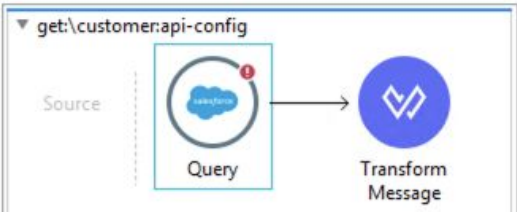
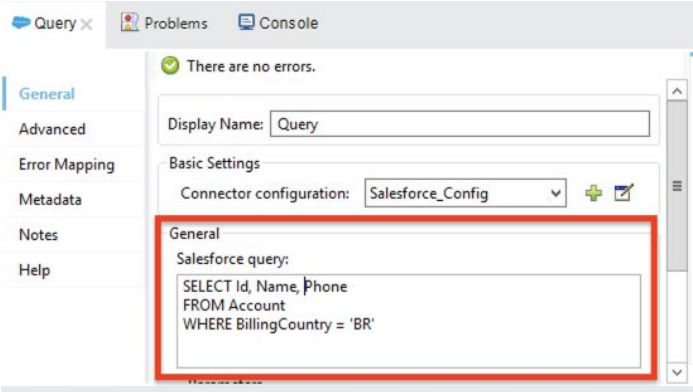
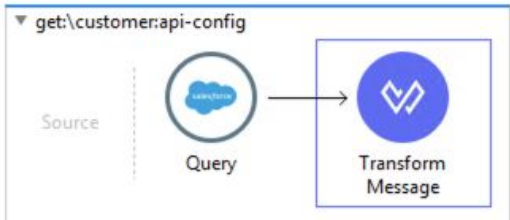
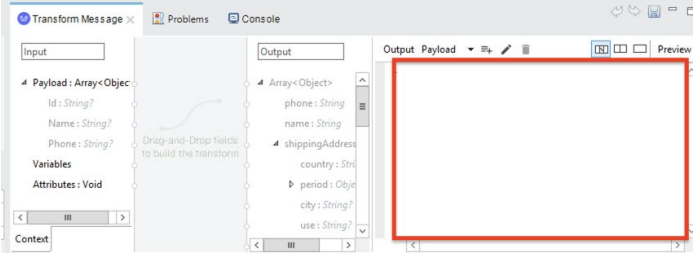
Etapa 5: Conectar-se ao Salesforce

Agora que você já possui o conector no fluxo de execução de sua aplicação, vamos definir as propriedades para realizar a conexão com o Salesforce.

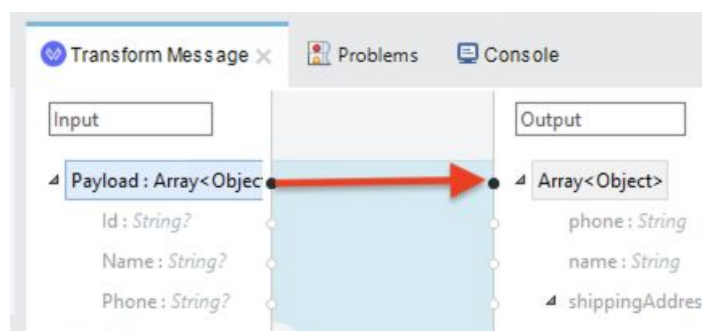
1. Clique no conector Salesforce Query	
2. No painel abaixo, observe as propriedades da consulta na aba Query Properties	
3. Clique no botão 	
4. A janela de Global Element Properties abrirá com informações do Salesforce Config	
5. Para a configuração do conector, colocaremos as seguintes propriedades: a. Username b. Password c. Security Token Importante: O instrutor do laboratório fornecerá as credenciais de acesso para essa etapa.	
6. Clique em [Test Connection...] para verificar se tudo está configurado corretamente.	
7. A conexão com Salesforce está funcionando!	

Etapa 6: Consultar os dados do objeto Account

É hora de consultar dados e encontrar as contas dos clientes no Salesforce

1. Clique no conector Salesforce Query .	
2. Escreva uma consulta SOQL para o Salesforce conforme exemplo: <pre>SELECT Id, Name, Phone FROM Account WHERE BillingCountry = 'BR'</pre>	
3. Exemplo de campos válidos no objeto Conta: <i>Name, BillingStreet, BillingCity, BillingState, BillingPostalCode, BillingCountry, BillingLatitude, BillingLongitude, BillingGeocodeAccuracy, Phone, AccountNumber</i> Observe que você deve especificar os campos porque não há opção "*" para selecionar todos os campos.	
4. Clique em Transform Message	
5. No painel abaixo, observe as propriedades na aba Transform Message Properties	
6. Exclua todo o conteúdo para que possamos substituí-lo por um novo mapeamento.	

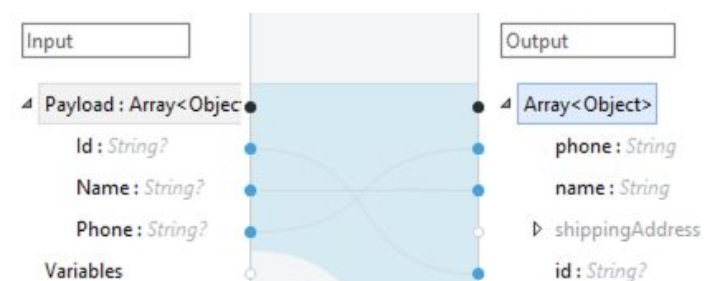
7. Comece a arrastar o **Payload: Array <Object>** para o correspondente **Array<Object>**



8. Em seguida, comece a mapear os demais campos:

- a. **Id** → **id**
- b. **Name** → **name**
- c. **Phone** → **phone**

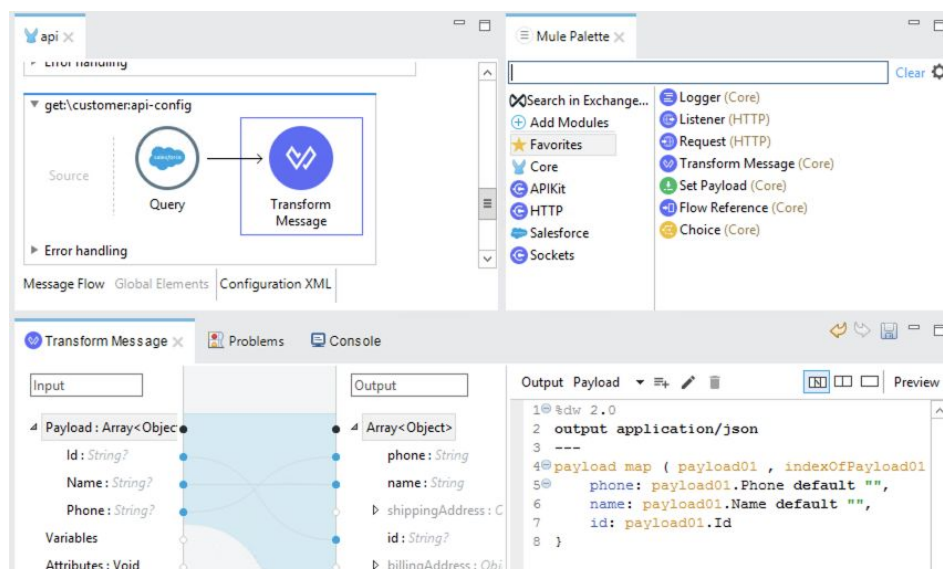
9. Adicione os mapeamentos necessários caso haja campos adicionais



10. Observe que o script de mapeamento é gerado automaticamente para você

```
1 @dw 2.0
2 output application/json
3 ---
4 payload map ( payload01 , indexOfPayload01 ) -> {
5   phone: payload01.Phone default "",
6   name: payload01.Name default "",
7   id: payload01.Id
8 }
```

Ao final desta etapa, temos um fluxo completo com consulta ao Salesforce.



Parabéns! Você criou uma consulta ao objeto Account no Salesforce e retornou os dados no formato JSON.

Etapa 7: Testar a API do Salesforce

Vamos explorar a aplicação Mule que acabou de ser criada.

1. Rode a aplicação com a integração Salesforce
2. Verifique na aba **Console** se o projeto aparece como "**DEPLOYED**".

```
* DEPLOYED *
```

```
*****
```

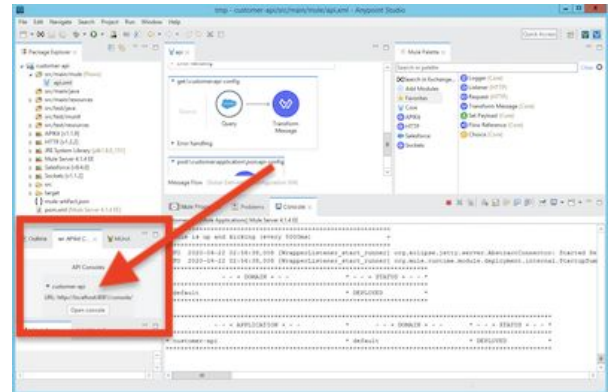
```
- - + DOMAIN + - - * - - + STATUS + - - *
```

```
*
```

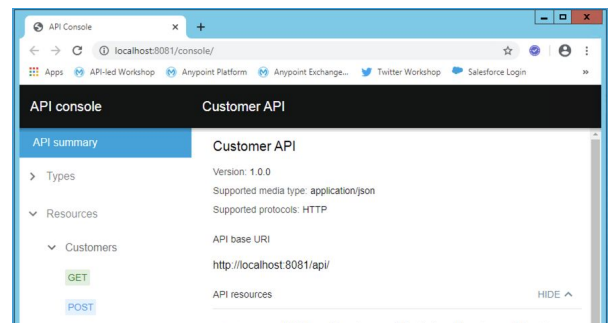
```
* default * DEPLOYED *
```

```
*****
```

3. Clique em **Console APIKit** → **Open console**

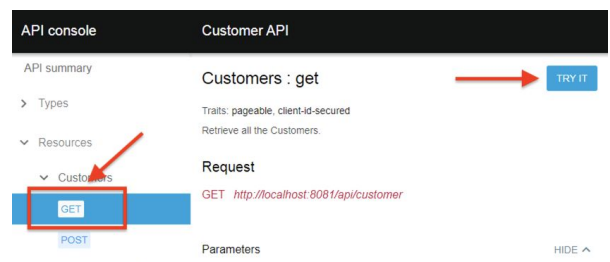


4. Uma janela do navegador abrirá com o endereço <http://localhost:8081/console/>



Vamos testar o fluxo de consulta **GET /customer** que acabamos de criar.

5. Clique no método **GET**.
6. Em seguida, aperte o botão **Try It** no canto superior direito do painel central



7. Clique no botão **Send**

Customer API

Request URL:

<http://localhost:8081/api/customer>

Parameters

Headers

Query parameters ☐ Show optional parameters

SEND

A solicitação retornará **200 OK**, seguida por uma lista das contas.

200 OK 849.49 ms



```
[Array[5]
  -0: {
    "phone": "1-(951)768-8479",
    "name": "Michelle Ruiz",
    "id": "0011Q00002EGFuaQAH"
  },
  ...]
```

Parabéns, a aplicação está rodando com sucesso! Em poucos minutos foi criada uma integração com Salesforce usando uma arquitetura moderna de API.

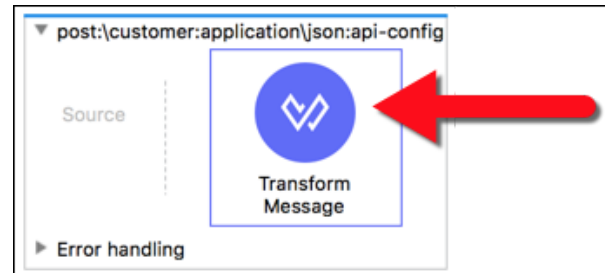
Etapa 8 (Opcional): Inserir um objeto no Salesforce

Nessa etapa, vamos implementar o método **POST /customer** usando o conector do Salesforce para inserir objetos no Salesforce.

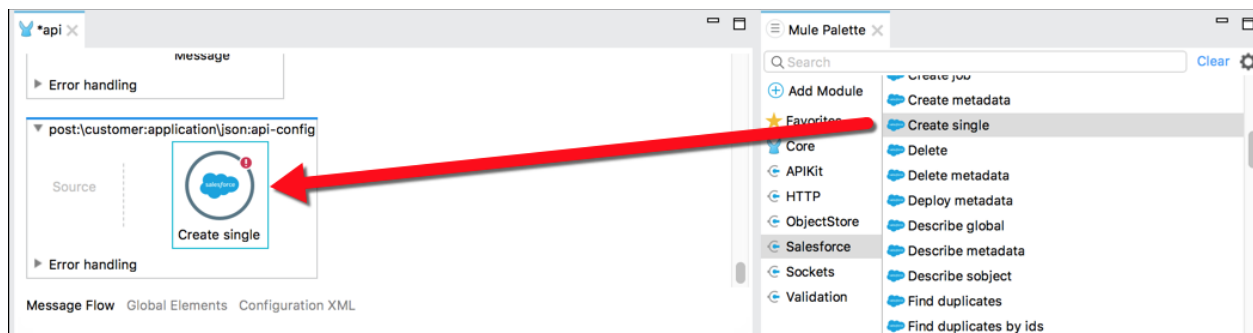
1. Localize o fluxo **post:\customer:application\json:api-config**

2. Exclua o operador **Transform Message**

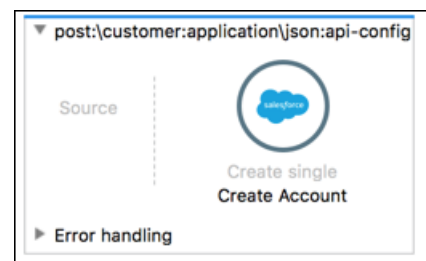
Nesse fluxo, o operador apenas simula a criação do objeto e retorna uma mensagem de sucesso. Nos próximos passos, substituiremos por uma operação Salesforce.



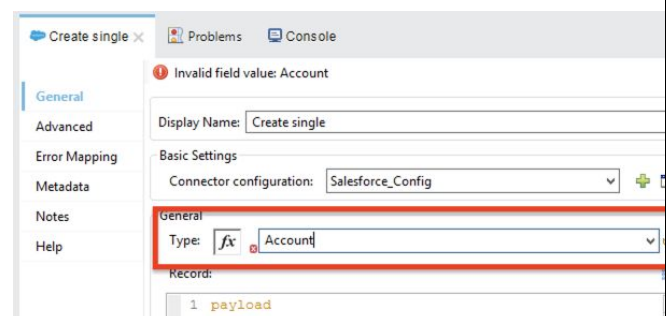
3. Em **Mule Palette**, selecione **Salesforce** e depois **Create Single**. Arraste e solte o componente no fluxo.



4. Clique no conector **Salesforce** e altere seu nome para **Create Account**

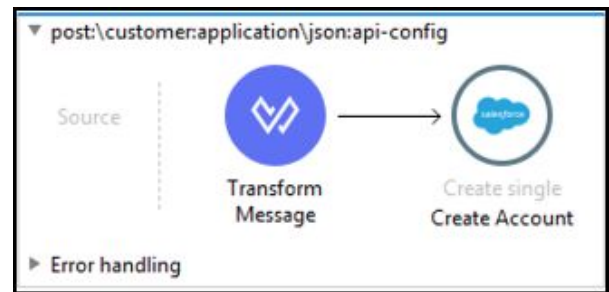


5. Em **Type**, selecione **Account**

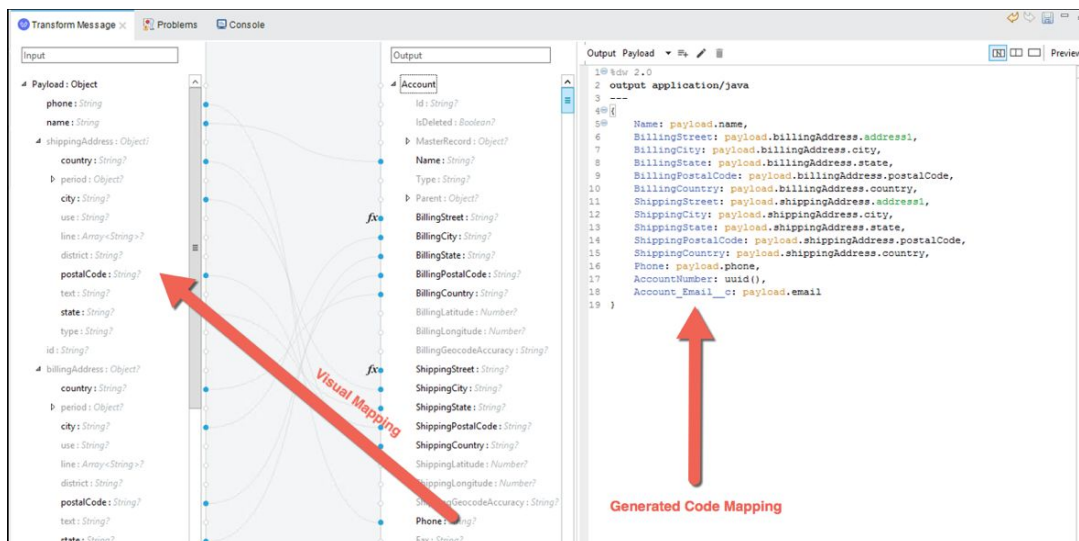


6. Arraste e solte um operador de **Transform Message** no início do fluxo, antes do conector Salesforce

Esse operador Transform será responsável por converter os parâmetros recebidos pela API para o formato do objeto Salesforce



Agora que configuramos o conector do Salesforce, precisamos criar um mapeamento que converta a mensagem JSON da solicitação HTTP no objeto **Account** do Salesforce.

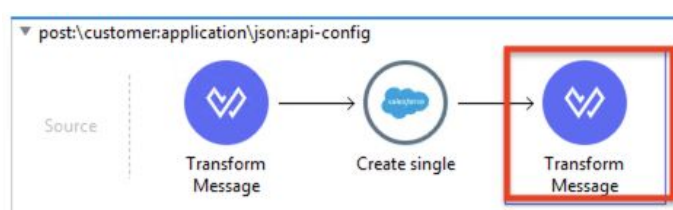


Este é o código de mapeamento que a imagem reflete. Para o nosso exercício de laboratório atual, verifique se os campos **Name**, **BillingCountry** e **BillingState** estão incluídos.

```
%dw 2.0
output application/java
---
{
    Name: payload.name,
    BillingCountry: payload.billingAddress.country,
    BillingState: payload.billingAddress.state,

    BillingStreet: payload.billingAddress.address1,
    BillingCity: payload.billingAddress.city,
    BillingPostalCode : payload.billingAddress.postalCode,
    ShippingStreet: payload.shippingAddress.address1,
    ShippingCity: payload.shippingAddress.city,
    ShippingState: payload.shippingAddress.state,
    ShippingPostalCode: payload.shippingAddress.postalCode,
    ShippingCountry: payload.shippingAddress.country,
    Phone: payload.phone,
    AccountNumber: uuid()
}
```

7. Arrasate e solte outro **Transform Message** e coloque-o no final do fluxo.

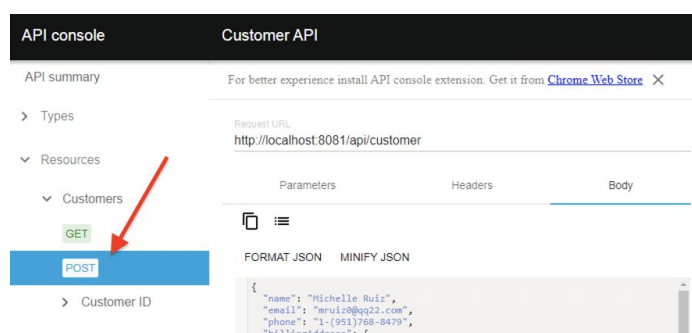


8. Nesta transformação, crie uma transformação muito simples

```
%dw 2.0
output application/json
---
payload
```

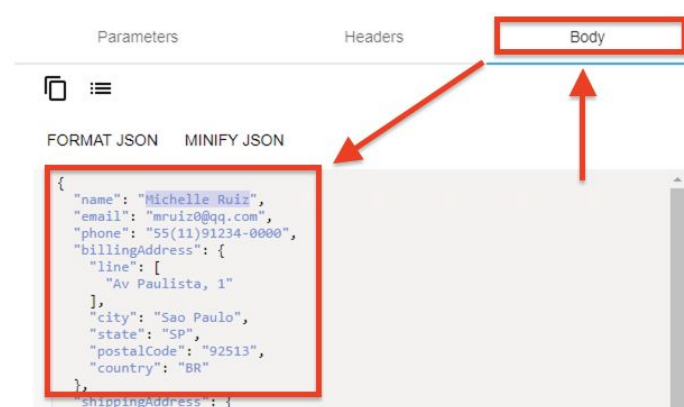
```
Output Payload
1 %dw 2.0
2 output application/json
3 ---
4 payload
```

9. Abra o **APIKit** → **Console** e selecione **POST**

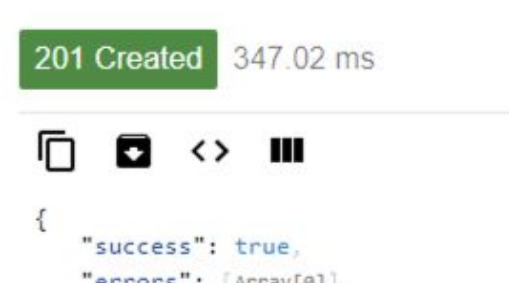


10. Clique em **Body** e, em seguida, modifique as informações de texto. Mude o nome, endereço, informações de contato.

Importante: Como estamos em um ambiente compartilhado de laboratório, não use seu email ou número de telefone real nessa etapa.

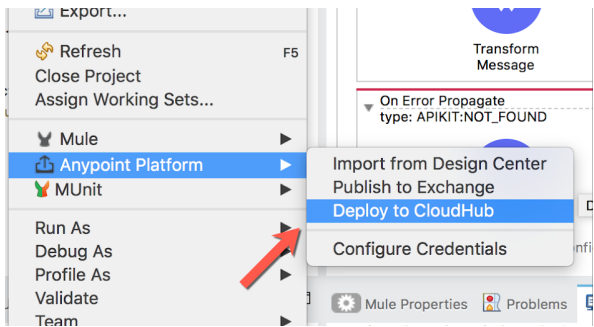
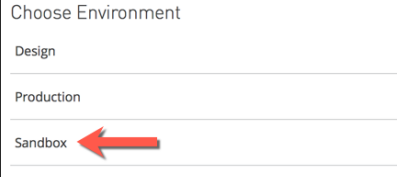


11. Clique no botão **Send** e verifique o resultado.

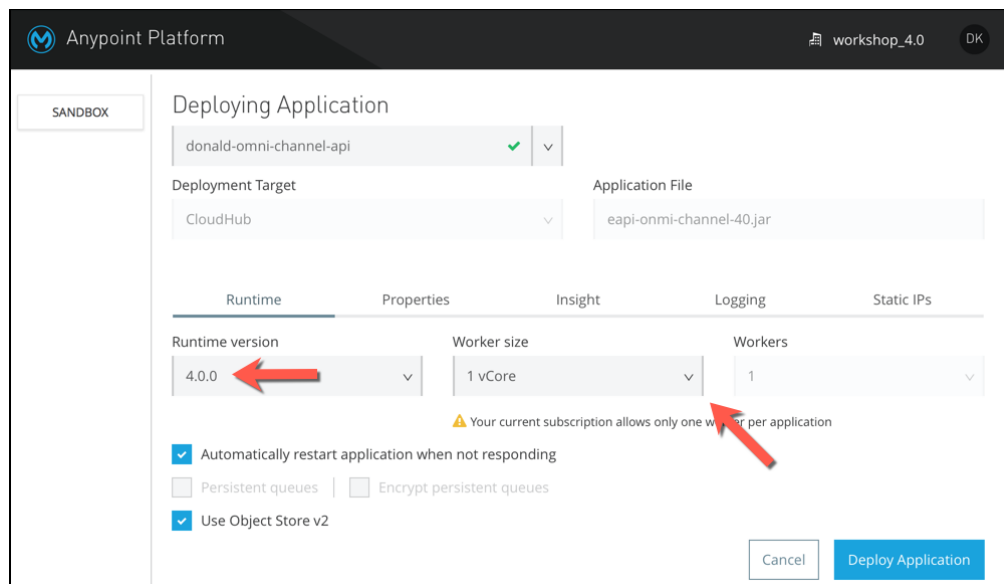


Etapa 9: Publicar a API no CloudHub

Vamos implantá-la no **CloudHub** para que a aplicação esteja disponível online.

<ol style="list-style-type: none">1. No painel à esquerda, procure o projeto api-customer no Package Explorer e clique com o botão direito do mouse.2. Selecione Anypoint Platform → Deploy to CloudHub	
<ol style="list-style-type: none">3. Selecione o ambiente Sandbox	

4. Definimos a configuração da aplicação para nossa API.



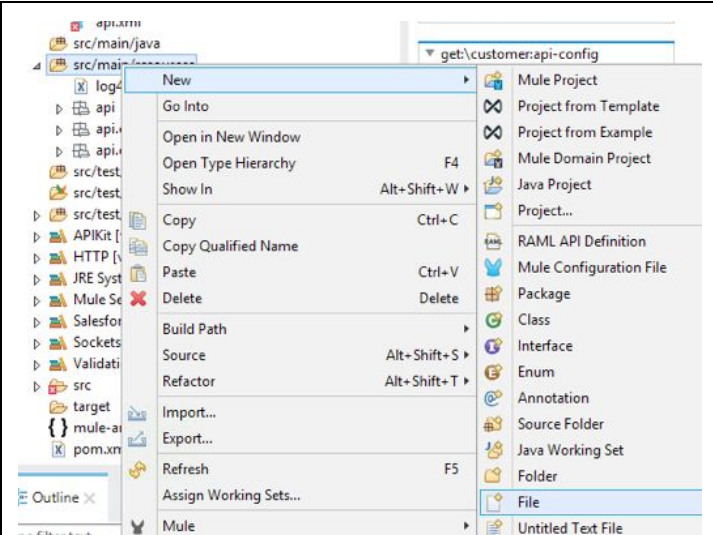
- a. No **nome da aplicação**, escolho um nome único para sua API (no exemplo acima, foi criado "*donald-omni-channel-api*"). Você verá uma verificação verde se o nome estiver disponível. Caso contrário, use outro nome.
 - b. Em **Worker size** selecione **0,1**.
5. Depois que a configuração estiver concluída, clique no botão **Deploy Application**.
 6. Depois que a implantação estiver concluída (pode levar alguns minutos), você poderá navegar no Console da API em <http://<nome-da-api>.<regiao-escolhida>.cloudhub.io/console/>

Etapa 10: Configurar arquivo de propriedades

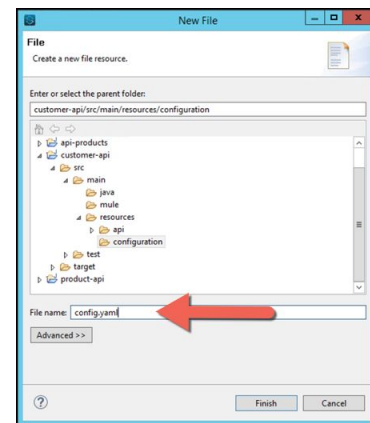
Nesta etapa, vamos criar um arquivo de propriedades para manter as credenciais de conexão com Salesforce separadas do código do projeto.

1. Crie um novo arquivo em **src/main/resources**

Nota: Neste passo inicial é comum criar o arquivo no diretório incorreto. Por exemplo: confundir a pasta **main** e **teste**, ou **mule** e **resources**. O local correto é **src/main/resources**.



2. Nomeie o arquivo para **config.yaml**


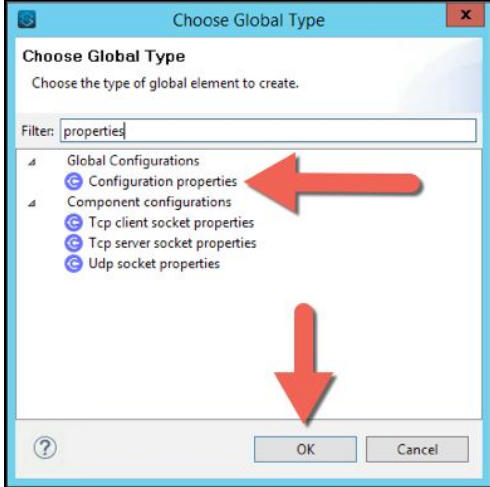



3. Coloque as informações de credenciais em **config.yaml**. Essas são as credenciais de conexão que sua API usará para o conector do Salesforce.

Importante: A sintaxe de arquivos YAML requer atenção nos espaçamentos e no uso de aspas

```
sfcdc:
  username: "demos+mythical_lab@mulesof
  password: "Elum1379"
  securityToken: "7ZDtkYMazEtbvgdaaPrtM"
```

Em seguida, associamos o arquivo de configuração **config.yaml** com o projeto **api-customer**.

<p>4. No painel central, retorne ao arquivo com fluxo Mule</p> <p>5. Clique na aba Global Elements</p>	
<p>6. Clique em Create</p> <p>7. Em Global Configurations, procure por Configuration properties e clique em OK</p>	
<p>8. Clique em reticências (...) e procure por config.yaml e clique em OK</p> <p><i>Nota: Embora seja fácil digitar o nome do arquivo, o ideal é clicar em (...) para confirmar que a existência do arquivo.</i></p>	

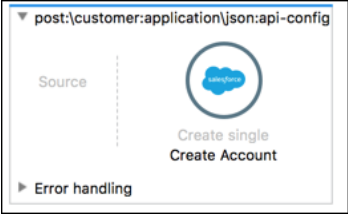

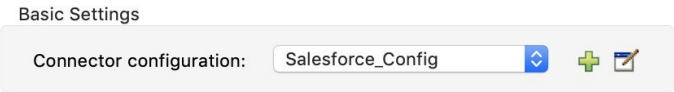
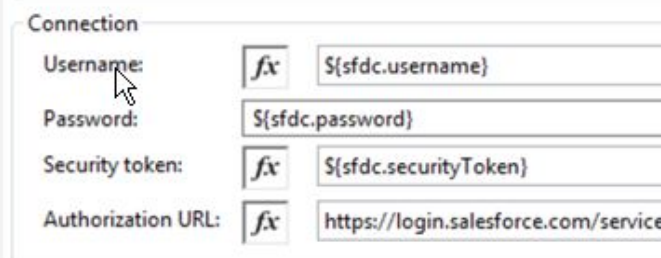
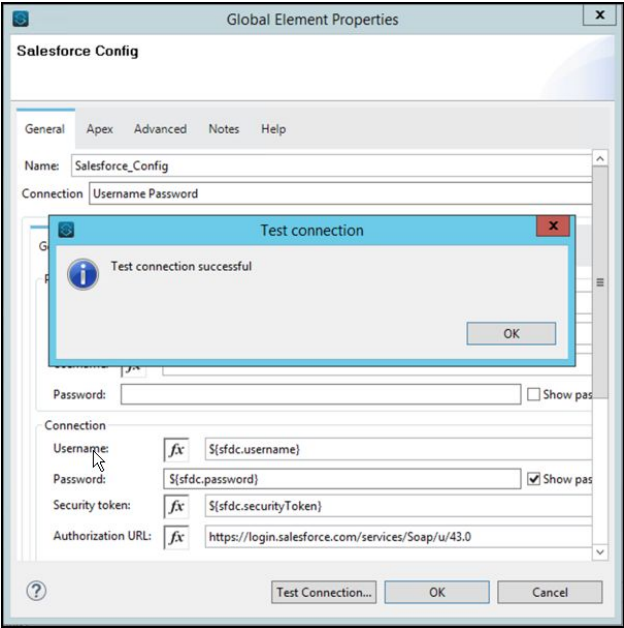
9. Clique no botão **Save All** para salvar as modificações.

A partir desse momento, todas as configurações podem ser feitas através de arquivos de configuração. Isso permite adotar diferentes propriedades em ambientes de desenvolvimento, homologação e produção.

Na próxima etapa, vamos remover as credenciais configuradas diretamente no conector e substituir pelo arquivo de propriedades.

Etapa 11: Proteger as credenciais do Salesforce

Agora que você tem seu arquivo de propriedade, vamos definir uma nova configuração para o conector.

<div>1. Volte para a Fluxo de mensagens em você usou o conector do Salesforce anteriormente e selecione-o.</div>	<div></div>
<div>2. Clique no botão  para editar a configuração do conector.</div>	<div></div>
<div>3. Para a configuração do conector vamos colocar as seguintes propriedades:<div>a. Username: <code>\${sfdc.username}</code></div><div>b. Password: <code>\${sfdc.password}</code></div><div>c. Security token: <code>\${sfdc.securityToken}</code></div></div>	<div></div>
<div>4. Clique em [Test connection...] para testar a conexão e verificar se tudo está configurado corretamente</div> <div>Parabéns! A partir desse momento, os conectores estão configurados com as credenciais presentes no arquivo de propriedade.</div> <div>Nota: caso haja problema no teste de conexão, a forma mais fácil de diagnosticar o problema é voltar à configuração inicial e ir substituindo as propriedades uma a uma.</div>	<div></div>