Implementando um provedor de pagamentos

A VTEX oferece diversas soluções para proporcionar uma experiência de Comércio Unificado aos clientes, sendo capaz de gerenciar pedidos tanto do e-commerce quanto das lojas físicas. Para a experiência na loja física, a VTEX oferece o aplicativo VTEX Sales , onde os vendedores podem atender os clientes e concluir todo o processo de venda. Ao finalizar a compra, o cliente terá a opção de efetuar o pagamento com um cartão físico em um terminal de pagamento, também conhecido como Ponto de Venda (PDV).

Para que os parceiros de pagamento integrem suas soluções de pagamento usando um PDV (Ponto de Venda), é necessário atender a uma série de requisitos. Você pode conferir um resumo desses requisitos na lista abaixo:

- Desenvolva um conector de pagamento usando o Protocolo de Provedor de Pagamento (PPP) . Existem alguns detalhes necessários no conector para processar pagamentos em um PDV (Ponto de Venda), tais como:
    - Inclua os métodos de pagamento aceitos (cartão de crédito ou débito) no Manifesto .
    - Utilize funções de retorno de chamada (callbacks) para pagamentos assíncronos.
    - Utilize os aplicativos de pagamento para identificar o terminal de ponto de venda (POS) e aguarde a resposta.
    - O endpoint deve retornar a qualquer chamada em menos de 20 segundos.
- Para fins de teste, configure uma loja VTEX com os métodos de pagamento suportados. Essa configuração deve ser fornecida pela VTEX.
- Caso seja necessário para testes, tenha um dispositivo com o aplicativo VTEX Sales instalado .
- Os desenvolvedores podem querer criar seu próprio aplicativo de pagamento para identificar o PDV (Ponto de Venda). Esse aplicativo também precisará ser instalado na loja.

Para desenvolver um novo conector de pagamento, é obrigatório seguir os pré-requisitos definidos pela VTEX . Você pode saber mais sobre eles na seção Pré-requisitos de implementação do nosso artigo sobre o Protocolo de Provedor de Pagamento .

Pré-requisitos do conector de pagamento

Existem algumas etapas necessárias para que o conector possa processar pagamentos no mundo físico, as quais são descritas abaixo:

1. Rota GET Listar Manifesto do Provedor de Pagamento (`/manifest`). O corpo da resposta do endpoint deve conter os métodos de pagamento `Venda Direta Credito`e `Venda Direta Debito`. Caso seja oferecida a opção de divisão de recebíveis, isso deve ser informado no `allowsSplit`campo para cada método de pagamento. Exemplo de corpo de resposta configurado corretamente:

```
{
 paymentMethods:
 [
    {
      name: "American Express",
      allowsSplit: "onCapture"
    },
    {
      name: "Diners",
      allowsSplit: "onCapture"
    },
    {
      name: "Elo",
      allowsSplit: "onCapture"
    },
```

```
    {
        name: "Hipercard",
        allowsSplit: "onCapture"
    },
    {
        name: "Mastercard",
        allowsSplit: "onCapture"
    },
    {
        name: "Visa",
        allowsSplit: "onCapture"
    },
    {
        name: "Boleto Bancário",
        allowsSplit: "onAuthorize"
    },
    {
        name: "Visa Electron",
        allowsSplit: "disabled"
    },
    {
        name: "Maestro",
        allowsSplit: "disabled"
```

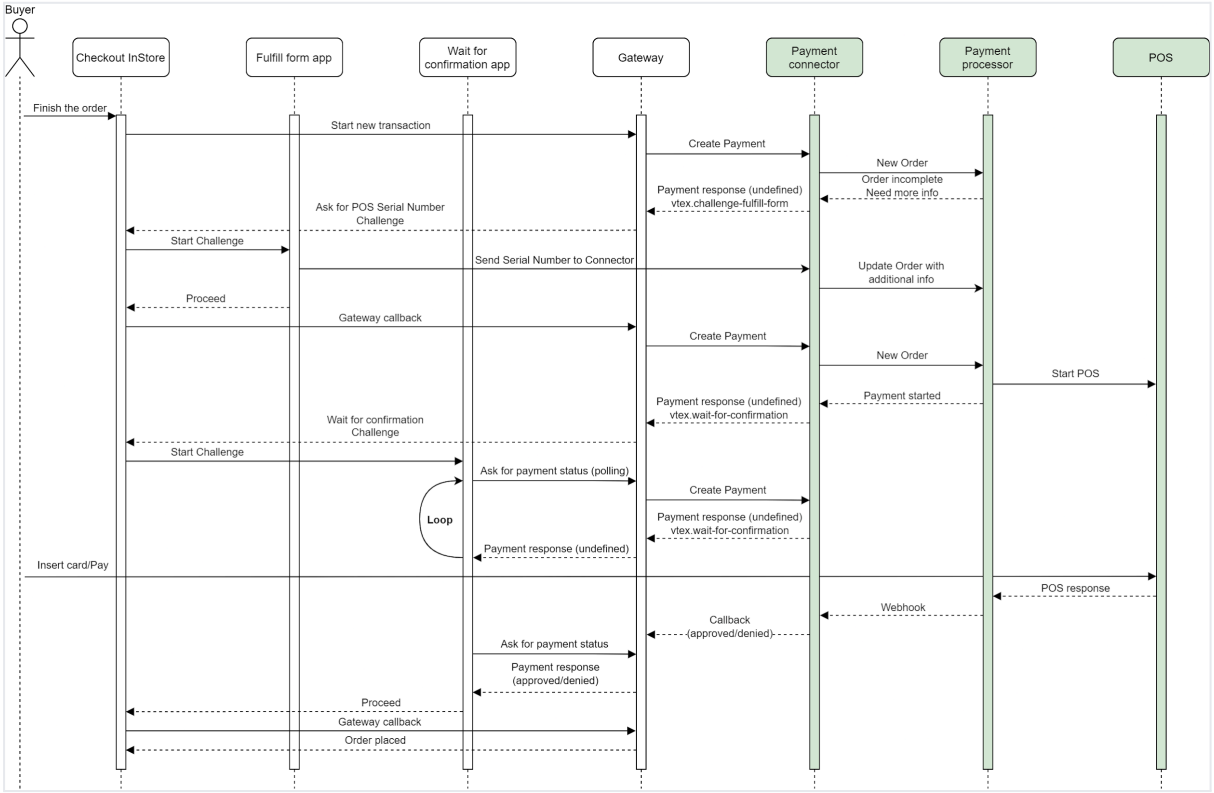
```
    },
    {
      name: "Pix",
      allowsSplit: "onAuthorize"
    },
+     {
+       name: "Venda Direta Debito",
+       allowsSplit: "onCapture"
+     },
+     {
+       name: "Venda Direta Credito",
+       allowsSplit: "onCapture"
+     }
  ]
}
```

2. Rota POST Create Payment (`/payments`). Esta rota deve ser idempotente, o que significa que o provedor deve ser capaz de lidar corretamente com chamadas repetidas da rota com o mesmo status `paymentId`. O provedor não deve recriar o pagamento para múltiplas chamadas, mas sim retornar o status mais atualizado.

Cenário e fluxo

Aqui descrevemos o fluxo de pagamento no contexto de uma loja física que utiliza um sistema de PDV (Ponto de Venda). O diagrama de sequência a seguir representa todas

as etapas desse fluxo, onde as barras verdes indicam as etapas pelas quais o provedor de pagamento é responsável:



1. O fluxo começa quando um comprador finaliza uma compra em uma loja física da VTEX criada no aplicativo de vendas da VTEX .
2. O aplicativo VTEX Sales faz uma solicitação de autorização ao gateway de pagamento VTEX.
3. Nosso gateway faz uma solicitação de criação de pagamento para o conector definido nas configurações da loja para o método de pagamento específico usado na compra (ou seja: `Venda Direta Credito`).
4. O Gateway recebe a resposta "Criar Pagamento" do conector com o `undefined`status e o `appname`campo do `paymentAppData`parâmetro preenchido com o nome de um aplicativo (ex: `vtex.challenge-terminal-connector-app`). Isso significa que o pagamento não foi concluído e precisa realizar uma verificação abrindo um Aplicativo de Pagamento , que será responsável por identificar o PDV.
5. O aplicativo VTEX Sales recebe a resposta do Gateway e abre o aplicativo para iniciar o desafio.
6. O vendedor interage com o aplicativo para identificar o terminal de ponto de venda (PDV) a ser usado no processo de pagamento. Em seguida, o aplicativo envia o número de série do PDV para o conector e encerra a conexão.

7. O conector solicita a atualização do pedido no processador de pagamentos com o número de série do PDV (Ponto de Venda).
8. Após o fechamento do aplicativo, um retorno de chamada é acionado no aplicativo VTEX Sales para o Gateway, iniciando a sequência de solicitações para efetuar o pagamento no PDV (Ponto de Venda).
    - O Gateway envia uma solicitação de Criação de Pagamento ao conector com o mesmo ID de pagamento usado na etapa 3. Isso significa que o Gateway está aguardando uma atualização sobre o status do pagamento.
    - O conector chama o processador de pagamentos para iniciar o pagamento no ponto de venda (PDV).
    - O processador de pagamentos chama o terminal de ponto de venda (POS) para efetuar o pagamento com o cartão. Em seguida, o processador de pagamentos retorna ao conector informando que o pagamento no POS foi iniciado.
    - Como leva algum tempo para o pagamento ser concluído no POS, o conector retorna ao Gateway mais uma vez com um `undefined`status e informações para abrir outro aplicativo de pagamento chamado `vtex.challenge-wait-for-confirmation`.
    - O Gateway retorna ao aplicativo VTEX Sales, instruindo-o a abrir o aplicativo "Aguardar confirmação" .
9. O aplicativo VTEX Sales abre a janela "Aguardar confirmação" , para que o usuário entenda visualmente que o aplicativo está aguardando a conclusão do pagamento pelo sistema de ponto de venda (PDV).
    - O aplicativo "Aguardar confirmação" solicita ao Gateway o status de pagamento atualizado.
    - O Gateway utiliza a solicitação Criar Pagamento para o conector, a fim de obter o status de pagamento atualizado.
    - Embora o sistema POS não finalize o pagamento e não responda, o Conector de Pagamento continua respondendo que o status do pagamento é `undefined`.
    - O Gateway responde ao aplicativo "Aguardar confirmação" com o `undefined`status.
    - O aplicativo "Aguardar confirmação" entra em um loop, repetindo a partir da etapa 9.1, chamando o Gateway novamente para obter o status atualizado até que o PDV finalize o pagamento e o status seja alterado ou ocorra um tempo limite. O tempo limite é definido no `secondsWaiting`parâmetro do payload. Se ocorrer um tempo limite, o pagamento é cancelado e o comprador precisa finalizar o pedido

novamente. Mais informações podem ser encontradas na subseção "Aplicativo aguardar confirmação" deste artigo.

10. Após a etapa 8.3, muitas das etapas ocorreram em segundo plano. Mas a partir desse ponto, o comprador pode inserir o cartão e interagir com o terminal de pagamento para efetuar o pagamento.
11. O POS responde ao processador de pagamentos a partir da solicitação na etapa 8.3.
12. O processador de pagamentos processa a transação e usa um webhook para chamar o conector.
13. O provedor de pagamento faz uma solicitação de retorno de chamada obrigatória ao Gateway (sem a qual não é possível aprovar ou rejeitar a transação POS), enviando as seguintes informações para a VTEX:
    - Carga útil contendo informações do cartão (campos `cardBrand`, `firstDigits`e `lastDigits`)
    - Status do pagamento (`approved`ou `denied`)
14. O aplicativo Aguardar confirmação solicita novamente ao Gateway o status de pagamento atualizado.
15. O aplicativo "Aguardar confirmação" recebe o status de pagamento atualizado.
16. O aplicativo "Aguardar confirmação" responde ao aplicativo de vendas VTEX informando que pode prosseguir, portanto, o aplicativo é fechado.
17. A partir de uma chamada de retorno, o aplicativo VTEX Sales entra em contato com o Gateway para verificar o status do pagamento.
18. O Gateway responde ao aplicativo VTEX Sales com o último status. Se o status do pagamento for "Em andamento" `denied`ou `canceled`"Não concluído", o aplicativo VTEX Sales exibe um erro e o comprador pode tentar finalizar a compra novamente no caixa. Caso contrário, a compra é concluída e o pedido é registrado no sistema.

Detalhes sobre a rota de Criação de Pagamento

O corpo da requisição do endpoint Criar Pagamento possui diversos parâmetros. Alguns deles requerem maior atenção no contexto de lojas físicas e são detalhados abaixo:

- `card`O gateway da VTEX não recebe dados de cartão para pagamentos em lojas físicas. Portanto, este objeto é enviado com todos os seus campos como `null`.

```
"card": {
        "holder": null,
```

```
        "number": null,
        "csc": null,
        "expiration": {
            "month": null,
            "year": null
        },
        "document": null,
        "token": null
    }
```


- `paymentMethod`Para pagamentos em lojas físicas, em vez de usar valores como `ecommerce` `"Visa"`e `fig` `"Mastercard"`, este campo será preenchido com os valores `ecommerce` `"Venda Direta Debito"`ou `fig` `"Venda Direta Credito"`. Se o conector funcionar tanto para pagamentos online quanto para pagamentos em lojas físicas, ele poderá determinar qual deles está sendo usado verificando este campo.
- `callbackUrl`Este campo é necessário, pois os pagamentos físicos seguem um fluxo assíncrono. O Gateway usa este campo para informar ao provedor qual URL deve ser usada para notificar a VTEX sobre atualizações no status do pagamento. Primeiro, o pagamento é criado com um `undefined`status. Em seguida, quando o status muda para `approved`ou `denied`, o provedor deve enviar uma solicitação para a URL do `callbackUrl`campo com o status atualizado. Você pode encontrar mais detalhes sobre esse fluxo na seção Interações adicionais e paymentAppData e no artigo Aplicativo de Pagamento .

Fluxo assíncrono e atraso para cancelar

Quando o VTEX Payment Gateway chama o endpoint Criar Pagamento, o conector inicia um processo de autorização assíncrono, que retorna imediatamente com um `undefined`status. Esse fluxo é semelhante a outros métodos de pagamento assíncronos, como Pix e Boleto (métodos comuns no Brasil). Após o Gateway receber o status inicial como `undefined`, em algum momento o pagamento será concluído e o status será atualizado para `approved`ou `denied`.

É importante observar que o pagamento não pode permanecer `undefined`ativo indefinidamente. Existe um limite de tempo que o Gateway da VTEX aguarda a conclusão do pagamento. Se esse limite de tempo for excedido, o Gateway chama automaticamente o endpoint Cancelar Pagamento do conector. Esse limite de tempo é definido pelo conector e informado pelo `delayToCancel`campo no corpo da resposta da rota Criar Pagamento, expresso em segundos.

Interações adicionais e dados de aplicativo de pagamento

Na VTEX, os pagamentos em lojas físicas seguem um fluxo assíncrono que requer interações adicionais com o usuário, também chamadas de desafios, até que sejam aprovadas. Para oferecer esse tipo de interação, o conector precisa informar ao nosso Gateway o `paymentAppData`campo na resposta do endpoint Criar Pagamento. Esse campo contém o nome do aplicativo VTEX IO (`appName`) que será apresentado no aplicativo VTEX Sales e os dados necessários para o funcionamento do aplicativo (`payload`).

Temos alguns aplicativos prontos para uso que permitem uma interação adicional com pagamentos em um ponto de venda (PDV).

Além dos aplicativos fornecidos, os parceiros também podem desenvolver aplicativos para suas necessidades específicas. Você pode conferir mais detalhes no artigo sobre Aplicativos de Pagamento .

Aplicativo de conector de terminal

Este aplicativo usa a câmera do dispositivo para ler o código de barras do terminal de ponto de venda (PDV) que será usado para a compra. Após a leitura do código de barras, as informações são enviadas automaticamente para a URL definida pelo `submitUrl`parâmetro. A carga útil enviada pelo aplicativo para a URL usa o seguinte formato: `{"serialNumber": "12345"}`.

Campos utilizados neste aplicativo:

- `appName`:
  - `vtex.terminal-connector-app`
- `payload`:
  - `submitUrl`(string): URL para onde a `serialNumber`mensagem será enviada.

Exemplo de carga útil:

```
"paymentAppData": {
    "appName": "vtex.terminal-connector-app",
    "payload":
        "{
            \"submitUrl\":
\"https://coolacquirer.com/instore_config/1231231\"
        }"
}
```

Aguarde a confirmação.

Este aplicativo é retornado pelo conector após o início do pagamento no PDV, permitindo que o aplicativo VTEX Sales verifique o status do pagamento. O tempo de espera para uma alteração no status do pagamento é definido pelo `secondsWaiting`parâmetro, que vem do conector na carga útil.

Campos utilizados neste aplicativo:

- `appName`:
  - `vtex.challenge-wait-for-confirmation`
- `payload`:
  - `secondsWaiting`(int): define por quantos segundos o aplicativo permanecerá aberto aguardando uma atualização do status do pagamento.

Exemplo de carga útil:

```
"paymentAppData": {
    "appName": "vtex.challenge-wait-for-confirmation",
    "payload": "{ \"secondsWaiting\": 600 }"
}
```

Interação com o sistema de ponto de venda (PDV)

Após o usuário interagir com o aplicativo que identifica o terminal de pagamento (POS), o processador de pagamentos é responsável pela comunicação com o POS para ler o cartão físico do comprador. Ao se comunicar com o POS, o processo de pagamento é realizado sem a intervenção da VTEX. O processador de pagamentos também é responsável por lidar com tentativas de pagamento decorrentes de erros de senha, saldo insuficiente e outros motivos.

Quando o processador de pagamentos recebe uma resposta informando se o pagamento foi aprovado ou negado, ele precisa comunicar ao Gateway de Pagamentos VTEX a resposta com o status do pagamento. Isso é feito por meio do fluxo de retorno de chamada, utilizando a URL no `callbackUrl`campo da solicitação de Criação de Pagamento. Você pode encontrar mais detalhes sobre esse fluxo na seção Autorização de Pagamento do artigo Protocolo do Provedor de Pagamento .

O fluxo de retorno de chamada começa com o conector de pagamento chamando a URL no `callbackUrl`campo para o Gateway. A solicitação de retorno de chamada pode ser de dois tipos: nova tentativa ou notificação. Se for uma nova tentativa, a `/retry`rota será usada e o Gateway fará outra solicitação de Criação de Pagamento para receber o status atualizado. Se for uma notificação, o conector chama o `/notification`endpoint para informar o Gateway sobre o status atualizado. Para pagamentos físicos, a carga útil da solicitação de retorno de chamada deve conter três campos adicionais sobre o

cartão usado no pagamento, que são utilizados para conciliações e relatórios financeiros do comerciante:

- `cardBrand`Marca do cartão.
- `firstDigits`Os seis primeiros dígitos do número do cartão.
- `lastDigits`Últimos quatro dígitos do número do cartão.

Testando seu conector

Antes que uma loja VTEX possa processar pagamentos no mundo físico, algumas etapas de configuração são necessárias no gateway de pagamentos. A VTEX fornece uma loja para o provedor de pagamentos testar durante o processo de desenvolvimento; essa loja já estará configurada e pronta para uso. No entanto, você pode seguir os passos abaixo para configurar uma loja sempre que necessário:

1. Instale o conector que você desenvolveu na loja. Se estiver criando um conector PPF usando o VTEX IO, consulte as etapas no artigo sobre o Payment Provider Framework para instalar corretamente esse tipo de conector. Caso precise de ajuda com a instalação, você pode abrir um chamado para a equipe de suporte da VTEX .
2. Configure uma afiliação de gateway com o conector instalado. Você pode encontrar mais detalhes no artigo "Registrando afiliações de gateway" .
3. Configure uma condição de pagamento que funcione com um POS. Para cartões de crédito, use "Venda Direta Crédito" e, para cartões de débito, " Venda Direta Débito" . Lembre-se de selecionar a afiliação configurada. Você encontra mais detalhes no artigo "Configurando condições de pagamento" .
4. Disponibilize o método de pagamento no aplicativo VTEX Sales seguindo os passos descritos no artigo "Definir métodos de pagamento exibidos no aplicativo VTEX Sales" . O ID das condições de pagamento utilizadas na `filter`matriz é `44`para débito e `45`para crédito e também pode ser consultado na página "Condições de Pagamento" do painel de administração.
5. Instale o aplicativo VTEX Sales no dispositivo de sua escolha.
6. Simule uma compra usando seu conector como condição de pagamento.

Disponibilizando seu conector para todos

Quando você decidir que o conector está pronto para ser disponibilizado em todas as lojas VTEX, poderá iniciar o processo de homologação. Será necessário testar os endpoints do seu conector com o aplicativo Payment Provider Test Suite e, caso todos os testes sejam aprovados, você deverá abrir um chamado para a equipe de suporte da VTEX para publicar o seu conector. Você pode encontrar mais detalhes sobre o processo de homologação nos artigos sobre o Protocolo de Provedor de Pagamento e a Homologação de Provedor de Pagamento .

Implementando um provedor de pagamentos

As seções a seguir mostram como desenvolver o middleware que interpretará as chamadas feitas entre o VTEX e o provedor.

Isso também mostra que o protocolo do provedor de pagamentos consiste em nove endpoints, divididos em duas sessões: Fluxo de pagamento e Fluxo de configuração . O middleware deve ser capaz de receber corretamente as informações fornecidas por todas essas chamadas.

Para desenvolver um novo conector de pagamento, é obrigatório seguir os pré-requisitos definidos pela VTEX . Você pode saber mais sobre eles na seção Pré-requisitos de implementação do nosso artigo sobre o Protocolo de Provedor de Pagamento .

Desenvolvendo o middleware

Requisitos

O middleware pode ser desenvolvido em qualquer linguagem de programação, sem restrições. No entanto, estabelecemos quatro requisitos para esta etapa:

- O endpoint deve ser servido via HTTPS na porta 443 com suporte a TLS 1.2. Esse formato é obrigatório; em hipótese alguma serão aceitos endpoints servidos via HTTP durante o processo de implementação.
- A API do parceiro deve ser pública. No processo de homologação, **não** aceitaremos nenhum tipo de API restrita.
- O parceiro deve criar um subdomínio ou um nome de domínio para o endpoint. Endereços IP não serão aceitos como nomes em nenhuma circunstância.
- Queremos que nossos processos sejam executados rapidamente, portanto, é importante que o endpoint responda aos testes de homologação em menos de 5 segundos. Além disso, o endpoint deve responder a qualquer chamada em menos de 20 segundos.

Com os requisitos alinhados, podemos prosseguir para o Fluxo de Pagamento.

Fluxo de pagamento

Os seguintes endpoints são usados para implementar as operações que ocorrem nos fluxos de autorização, cancelamento ou reembolso e liquidação.

1. Obter manifesto

Por meio desse endpoint, o manifesto do provedor é exibido, consistindo em diversas configurações de metadados, como métodos de pagamento, configuração de divisão e campos personalizados.

```
{
  "paymentMethods": [
    {
      "name": "Visa",
      "allowsSplit": "onCapture"
    },
    {
```

```
      "name": "Pix",
      "allowsSplit": "disabled"
    },
    {
      "name": "MasterCard",
      "allowsSplit": "onCapture"
    },
    {
      "name": "American Express",
      "allowsSplit": "onCapture"
    },
    {
      "name": "BankInvoice",
      "allowsSplit": "onAuthorize"
    },
    {
      "name": "Privatelabels",
      "allowsSplit": "disabled"
    },
    {
      "name": "Promissories",
      "allowsSplit": "disabled"
    }
```

```
    ]

  }
```

Caso ofereça suporte a algum tipo de pagamento personalizado , indique apenas o tipo de método suportado (Cobranded, Privatelabels ou Promissories) no `"name"`campo. Não é necessário descrever o nome específico do pagamento personalizado suportado (por exemplo, Promissory do Banco Colombiano).

Para obter mais informações sobre esta etapa, você pode consultar a documentação do endpoint GET List Payment Provider Manifest .

2. POST Criar Pagamento

Quando um cliente efetua uma compra no checkout, o provedor deve criar um pagamento relacionado ao pedido. Portanto, o provedor deve estar preparado para receber o corpo da requisição que conterá todas as informações do carrinho:

```
{

   "reference": "32478982",

   "orderId": "1047232106697",

   "selerId": "xpto",

   "transactionId": "611966",

   "paymentId": "5B127F1E0C944EF9ACE264FEC1FC0E91",

   "paymentMethod": "cash",

   "paymentMethodCustomCode": "{{paymentMethodCustomCode}}",

   "merchantName": "lojaexemplo",

   "value": "29,90",

   "currency": "BRL",

   "installments": "0",

   "deviceFingerprint": "12ade389087fe",

   "card": {
```

```
        "holder": "{{cardHolder}}",
        "number": "{{cardNumber}}",
        "csc": "{{cardSecurityCode}}",
        "expiration": {
            "month": "{{cardExpirationMonth}}",
            "year": "{{cardExpirationYear}}"
        }
    },
    "miniCart": {
        "shippingValue": 11.44,
        "taxValue": 10.01,
        "buyer": {
            "id": "c1245228-1c68-11e6-94ac-0afa86a846a5",
            "firstName": "John",
            "lastName": "Doe",
            "document": "01234567890",
            "documentType": "CPF",
            "email": "john.doe@example.com",
            "phone": "+12125357200"
        },
        "shippingAddress": {
            "country": "USA",
            "street": "12 E 49th St",
```

```
            "number": "10017",
            "complement": "Tower 49 Gallery",
            "neighborhood": "Central Midtown",
            "postalCode": "10017",
            "city": "New York",
            "state": "NY"
        },
        "billingAddress": {
            "country": "USA",
            "street": "12 E 49th St",
            "number": "10017",
            "complement": "Tower 49 Gallery",
            "neighborhood": "Central Midtown",
            "postalCode": "10017",
            "city": "New York",
            "state": "NY"
        },
        "items": [
            {
                "id": "132981",
                "name": "My First Product",
                "price": 2134.90,
                "quantity": 2,
```

```
                "discount": 5.00

            },

            {

                "id": "123242",

                "name": "My Second Product",

                "price": 21.98,

                "quantity": 1,

                "discount": 1.00

            }

        ]

    },

    "url": "https://admin.mystore.example.com/orders/v32478982",

    "callbackUrl": "{{callbackUrl}}",

    "returnUrl": "{{returnUrl}}"

}
```

Este `callbackURL`é um URL usado para realizar novas tentativas no fluxo da operação. Além disso, se o parceiro adotar o redirecionamento como fluxo de pagamento, você também deverá incluir o URL `returnURL`no payload da solicitação. Dessa forma, a VTEX poderá realizar o fluxo de redirecionamento, se necessário.

Ambos, ` `callbackURL`and` e `or`, `returnURL`são desenvolvidos pelo fornecedor.

Por outro lado, precisamos que seu provedor retorne esta chamada com a seguinte resposta:

```
{

    "paymentId": "5B127F1E0C944EF9ACE264FEC1FC0E91",
```

```
   "status": "undefined",

   "authorizationId": "AUT-E4B9C36034-ASYNC",

   "paymentUrl":
"https://exemplo2.vtexpayments.com.br/api/pub/fake-payment-provider/pay
ment-redirect/611966/payments/5B127F1E0C944EF9ACE264FEC1FC0E91",

   "nsu": "NSU-171BE62CB7-ASYNC",

   "tid": "TID-20E659E8E5-ASYNC",

   "acquirer": "TestPay",

   "code": "2000-ASYNC",

   "message": null,

   "delayToAutoSettle": 21600,

   "delayToAutoSettleAfterAntifraud": 1800,

   "delayToCancel": 21600

}
```


Os parâmetros `timeout` `delayToAutoSettle`e `delayToAutoSettleAfterAntifraud`cancel` definem o tempo total que o sistema aguardará para iniciar o fluxo de captura. Um processo semelhante ocorre para o parâmetro `cancel` `delayToCancel`, mas após o tempo limite, o sistema prosseguirá para o fluxo de cancelamento.

Todos esses três períodos de tempo são estabelecidos pelo provedor e devem ser contados em segundos.

Além disso, os parâmetros `code`e `message`são campos para os quais o provedor pode nos enviar qualquer informação sobre o processo. Ambos são opcionais e aparecerão nas próximas chamadas.

Todos os campos estão descritos na documentação POST Create Payment .

3. Cancelar pagamento

Para cancelar um pagamento, você precisa tê-lo criado previamente. Seu provedor precisa estar preparado para receber apenas duas informações: o `paymentId`, o

identificador do pagamento que será cancelado, e o `requestId`, o identificador que garante sua idempotência.

Por “idempotência”, entendemos a capacidade de uma ação ser realizada várias vezes. Como explicamos na seção Operações , o fluxo de cancelamento pode ser repetido durante todo o dia até que o provedor responda à solicitação.

```
{

   "paymentId": "F5C1A4E20D3B4E07B7E871F5B5BC9F91",

   "requestId": "1234"

}
```

Após o fornecedor processar o cancelamento do pagamento, esperamos uma resposta como esta:

```
{

   "paymentId": "F5C1A4E20D3B4E07B7E871F5B5BC9F91",

   "message": "Successfully cancelled",

   "code": null,

   "cancellationId": "1457BD07E6",

   "requestId": "1234"

}
```

Consulte a documentação completa para obter mais detalhes.

4. Pagamento de captura pós-venda

Se a sua transação for concluída com sucesso, o provedor poderá capturar o pagamento. Aliás, é muito importante que você saiba que a VTEX suporta capturas parciais.

Isso acontece porque o pagamento pode ser registrado em dois momentos diferentes:

- Quando a loja emitir a fatura do pedido.
- Após o período definido nos campos `delayToAutoSettle`, `delayToAutoSettleAfterAntifraud`ocorre o tempo limite.

Nesse contexto, o campo `value`é obrigatório e crucial para que a operação seja realizada corretamente.

A VTEX enviará as informações abaixo para processar o pagamento:

```
{

   "paymentId": "5B127F1E0C944EF9ACE264FEC1FC0E91",

   "transactionId": "611966",

   "value": 20.0,

   "requestId": "5678"

}
```

A resposta será semelhante ao corpo da solicitação. No entanto, o parâmetro `value`pode ser atendido com o valor total enviado na solicitação ou com um valor menor. Tudo depende de quanto o provedor deseja capturar.

```
{

   "paymentId": "5B127F1E0C944EF9ACE264FEC1FC0E91",

   "settleId": "CEE16492C6",

   "value": 20.0,

   "code": null,

   "message": null,

   "requestId": "5678"

}
```

Consulte a documentação do endpoint de Captura de Pagamento para obter mais detalhes.

5. Pagamento de reembolso POST

A VTEX também oferece suporte a reembolsos parciais, seguindo uma lógica semelhante à da captura.

Por exemplo, considere um carrinho de compras com dois itens. Um cenário possível é que a loja precise devolver apenas um dos itens comprados. Nesse caso, podemos

reembolsar apenas o valor desse único item, em vez de reembolsar o valor total, que inclui os dois itens e o frete.

Em qualquer caso, o fornecedor deve estar preparado para receber o seguinte pedido:

```
{

   "paymentId":
"VQKIIBUVOFDBIDLKZPOWSKETDYWCMJSACDVXWFCJVSKXGYVBBVISZRJLLQEKERJEMDYEIN
OUMFAZZGNEDVBQBABLUKLFBSEEIGLCAQTOGOGURKLFCAHJQTDMBNKYBIST",

   "transactionId": "611966",

   "settleId": "31018A3281",

   "value": 10.0,

   "requestId": "5678"

}
```

A resposta esperada é:

```
{

   "paymentId":
"VQKIIBUVOFDBIDLKZPOWSKETDYWCMJSACDVXWFCJVSKXGYVBBVISZRJLLQEKERJEMDYEIN
OUMFAZZGNEDVBQBABLUKLFBSEEIGLCAQTOGOGURKLFCAHJQTDMBNKYBIST",

   "refundId": null,

   "value": 0.0,

   "code": "refund-manually",

   "message": "Refund should be done manually",

   "requestId": "5678"

}
```

Lembre-se que, se necessário, o prestador de serviços pode retornar a chamada com o campo `value`preenchido com um valor inferior ao informado na solicitação para efetuar um reembolso parcial.

Consulte a página de finalização "Pós-pagamento de reembolso" para obter mais detalhes.

6. Solicitação POST de entrada (BETA)

A Solicitação de Entrada (BETA) implementa um URL que facilita a conexão direta entre o nosso serviço de Gateway e o Provedor de Pagamento.

É responsável pela comunicação com o sistema de backend do provedor através do nosso gateway, contando com a segurança do envio do contexto da transação pela VTEX.

Um exemplo da solicitação é:

```
{

   "requestId": "LA4E20D3B4E07B7E871F5B5BC9F91",

   "transactionId": "D3AA1FC8372E430E8236649DB5EBD08E",

   "paymentId": "F5C1A4E20D3B4E07B7E871F5B5BC9F91",

   "authorizationId": "{{authorizationId}}",

   "tid": "{{tid}}",

   "requestData": {

       "body": "{{originalRequestBody}}"

   }

}
```

Uma resposta esperada seria:

```
{

 "requestId": "LA4E20D3B4E07B7E871F5B5BC9F91",

 "paymentId": "F5C1A4E20D3B4E07B7E871F5B5BC9F91",

 "responseData": {

   "statusCode": 200,

   "contentType": "application/json",
```

```
   "content": "{\"myAttribute\":\"anyValue\"}"

 }

}
```

Consulte o endpoint Post Inbound Request (BETA) para obter mais detalhes.

Fluxo de configuração

Na VTEX, o comerciante precisa habilitar um ou mais conectores através do painel de administração. A configuração é basicamente a mesma para todos os parceiros e você pode consultar o tutorial em nossa Central de Ajuda.

No entanto, essa operação segue um fluxo de autenticação que permite ao provedor reconhecer e autorizar a solicitação do comerciante. Os seguintes endpoints geram três credenciais — `appKey`, `appToken`e `applicationId`— que são salvas pelo sistema VTEX, o que permite ao comerciante habilitar o conector sem precisar copiar e colar nenhuma credencial.

Este fluxo é opcional e não afeta sua conexão com a VTEX. Se você não pretende implementá-lo, pule para a seção de Homologação .

1. POST Criar Token de Autorização

Na primeira etapa, o corpo da solicitação será:

```
{

   "applicationId": "vtex",

   "returnUrl":
"https://admin.mystore.example.com/provider-return?authorizationCode="

}
```

O `applicationId`"Vtex" será sempre "vtex".

Em seguida, o provedor deve retornar um token de identificação para que possamos redirecionar o comerciante para o aplicativo do provedor.

```
{
```

```
   "applicationId": "vtex",

   "token": "auth_token_39766d98535d43a491d03b8c3bea060f"

}
```

Consulte a documentação do endpoint Criar Token de Autorização para verificar todos os detalhes.

2. Autenticação do provedor GET

Essa chamada permite ao comerciante analisar os termos e condições do fornecedor para usar o serviço.

Com o token fornecido no corpo da resposta anterior, a VTEX pode redirecionar o comerciante para a página de login do provedor.

No corpo da resposta, esperamos uma `authorizationCode`informação relacionada à que `returnUrl`foi enviada a você na primeira etapa do fluxo de configuração.

```
{

   "authorizationCode": "auth_code_5b7be276c8e04e95bb1e",

   Grant access to "vtex" application

}
```

Consulte o endpoint Obter Autenticação do Provedor para obter mais detalhes.

3. OBTER Credenciais

Por fim, o provedor deve nos enviar os valores das três credenciais.

Esperamos uma resposta semelhante a esta:

```
{

   "applicationId": "vtex",

   "appKey": "test_key_AE06E97A8C5B45DFA2DC665D6BE91E",

   "appToken": "test_token_90FB36380D114B37BC0557AEEE40ED"

}
```

Dessa forma, as credenciais serão salvas em nosso sistema e ativadas no momento em que o comerciante decidir habilitar o conector.

Consulte o endpoint Obter Credenciais para verificar todos os detalhes.

Implementando um provedor de pagamentos

O Payment Provider Framework (PPF) é uma forma alternativa de desenvolver conectores de pagamento para sua integração com o VTEX IO. Como o desenvolvimento é baseado em um modelo de aplicativo IO, grande parte do trabalho já está concluída para implementar os recursos necessários, incluindo as rotas da API , os tipos usados nos corpos de requisição e resposta e o Proxy Seguro . Com o PPF, os desenvolvedores também não precisam se preocupar com a hospedagem do conector, pois ele é hospedado na infraestrutura do IO.

Para desenvolver um novo conector de pagamento, é obrigatório seguir os pré-requisitos definidos pela VTEX . Você pode saber mais sobre eles na seção Pré-requisitos de implementação do nosso artigo sobre o Protocolo de Provedor de Pagamento . Para obter mais informações sobre como integrar provedores de pagamento à VTEX, visite Integrando um novo provedor de pagamento à VTEX .

Começando

Clonagem do repositório base

Se você estiver iniciando um projeto totalmente novo, recomendamos que clone o repositório de exemplo , pois ele já possui todas as configurações básicas definidas.

Atualizando seu projeto

Depois de ter o código do repositório em seu espaço de trabalho, você precisa garantir que possui todas as dependências necessárias e que elas estejam atualizadas. Para isso, siga estes passos:

1. Execute o seguinte comando na pasta do seu nó:

```
yarn add @vtex/payment-provider
```

2. Acesse suas configurações `package.json`e verifique se o programa foi adicionado como uma dependência com a versão correta:

```
"@vtex/payment-provider": "1.x"
```

3. Verifique a `package.json`versão de `@vtex/api`, que deve estar listada em devDependencies da seguinte forma:

```
"@vtex/api": "6.x"
```

4. Ao vincular seu aplicativo, esta versão pode ser atualizada para uma versão posterior à 6.x, o que é normal. Caso não esteja listada como uma versão compatível `devDependency`, execute o seguinte comando na sua pasta do Node:

```
yarn add -D @vtex/api
```

Se você encontrar erros de tipo ou conflitos no seu projeto relacionados a `@vtex/api`, siga estes passos: 1. Exclua a `node_modules`pasta e o `yarn.lock`arquivo tanto da raiz do seu projeto quanto da pasta do Node do seu projeto. 2. Execute o comando `yarn install -f`em ambas as pastas.

5. Em seu arquivo .bashrc `manifest.json`, você deve verificar a seção de construtores, na qual você deve incluir o arquivo .bashrc `paymentProvider`em sua versão atual. Isso adicionará políticas para fazer callbacks nas APIs do Gateway de Pagamento e também exporá as rotas do protocolo do Provedor de Pagamento.

```
"builders": {

 "node": "6.x",

 "paymentProvider": "1.x"

}
```

Próximos passos

Para criar seu serviço, você deve implementar o conector do provedor de pagamentos e o próprio serviço , conforme descrito nas seções abaixo.

Provedor de Pagamento

Esta é uma classe abstrata com as assinaturas das funções de roteamento necessárias em seu conector, de acordo com o protocolo .

Você deve criar uma nova classe que estenda a classe `Payment Provider`, a qual deve implementar uma função para cada rota. As funções receberão o corpo da requisição (quando houver) como parâmetro e a resposta deverá ser retornada como um objeto, como no exemplo abaixo:

```
import {

PaymentProvider,

// ...

} from '@vtex/payment-provider'



class YourPaymentConnector extends PaymentProvider {



// ... implementation of the other routes functions

}
```

O Typescript deve verificar automaticamente erros de digitação, mas, se necessário, você pode verificar as assinaturas de solicitações e respostas dos endpoints do Fluxo de Pagamento em nossa Referência da API .

Construtor de provedores de pagamento

Para especificar quais métodos de pagamento o conector aceitará processar, siga os passos abaixo:

1. Crie uma pasta com o nome `paymentProvider`usando a seguinte estrutura de pastas.

```
node

paymentProvider

manifest.json
```

2. Crie um arquivo com o nome `configuration.json`dentro da `paymentProvider`pasta.

```
node

paymentProvider

  |--configuration.json

manifest.json
```

3. Declare os métodos de pagamento aceitos pelo seu provedor de pagamento. Essa ação permite que eles sejam implementados automaticamente pelo construtor, sem a necessidade de declará-los na `/manifest`rota.

```
{

 "name": "MyConnector",

 "paymentMethods": [

   {

     "name": "Visa",

     "allowsSplit": "onCapture"

   },

   {

     "name": "American Express",

     "allowsSplit": "onCapture"

   },

   {

     "name": "Diners",

     "allowsSplit": "onCapture"

   },
```

```
    {
      "name": "Elo",
      "allowsSplit": "onCapture"
    },
    {
      "name": "Hipercard",
      "allowsSplit": "onCapture"
    },
    {
      "name": "Mastercard",
      "allowsSplit": "onCapture"
    },
    {
      "name": "BankInvoice",
      "allowsSplit": "onAuthorize"
    }
  ]
}
```

O `name`campo que indica o nome do conector deve ser substituído pelo nome do seu provedor. Este campo não pode ser preenchido com um valor `"MyConnector"`.

Para verificar quais métodos de pagamento estão disponíveis no painel de administração da VTEX, acesse Configurações da Loja > Pagamento > Configurações ou digite "Configurações" na barra de pesquisa na parte superior da página e clique no `+`botão verde. Se você deseja adicionar um novo método de pagamento que ainda não está disponível, abra um chamado para a equipe da VTEX informando sobre esse novo método de pagamento.

Você também pode declarar o `customFields`array para permitir que seu provedor de pagamento envie informações específicas. O `type`campo pode ser configurado da seguinte forma: `text`para dados não confidenciais; `password`para informações sensíveis e de segurança (exceto `appKey`as `AppToken`que não devem ser enviadas neste campo); e `select`para agrupar um conjunto de informações personalizadas.

```
{
 "name": "MyConnector",
 "paymentMethods": [
   ...
 ],
 "customFields": [
   {
     "name": "Company account",
     "type": "text"
   },
   {
     "name": "POS URL",
     "type": "text"
   },
   {
     "name": "Client key",
     "type": "password"
   },
   {
     "name": "Auto Capture Settings",
```

```
      "type": "select",
      "options": [
        {
          "text": "Automatic Capture Immediately After Payment
Authorization",
          "value": "Immediately"
        },
        {
          "text": "Auto Settle Delay: 7 Days",
          "value": "Deactivated"
        }
      ]
    }
 ]
}
```

Substituindo a rota do Manifesto

Caso deseje alterar a `/manifest`rota padrão devido a alguma funcionalidade específica do seu provedor, abra um chamado para a equipe de suporte da VTEX descrevendo seu caso de uso e adicionando os parâmetros especiais conforme especificado abaixo.

```
{
 "memory": 256,
 "ttl": 10,
 "timeout": 10,
 "minReplicas": 2,
 "maxReplicas": 3,
```

```
"routes": {

  "manifest": {

      "path": "/_v/api/my-connector/manifest",

      "handler": "vtex.payment-gateway@1.x/providerManifest",

      "headers": {

        "x-provider-app": "$appVendor.$appName@$appVersion",

      },

      "public": true

    }

 }

}
```

O `x-provider-app`parâmetro deve ser atualizado sempre que houver uma alteração importante (por exemplo, vtex.payment-provider-example@1.2.3). Você também pode omitir os parâmetros `handler`and` `headers`, porém, ao executar este procedimento, você precisará implementá-los manualmente.

Opções configuráveis disponíveis

Além dos campos do manifesto (`paymentMethods`e `customFields`), as seguintes opções de configuração estão disponíveis:

Nome do parâmetro	Obrigatório	Padrão	Descrição
`name`	Sim		Nome do conector do provedor de pagamento.
`serviceUrl`	Sim	Gerado automaticamente para	Uma URL válida (pode incluir caminhos relativos).

		conectores de E/S.	
`implementsOAuth`	Não	`false`	Define se o provedor implementa o fluxo de configuração com suporte à autenticação OAuth.
`implementsSplit`	Não	`false`	Define se o provedor implementa o fluxo de divisão de pagamentos.
`usesProviderHeadersName`	Não	`true`	Define se o provedor receberá os cabeçalhos appKey e appToken como `"x-provider-api-appKey"`e `"x-provider-api-appToken"`respectivamente.
`useAntifraud`	Não	`false`	Define se os provedores de serviços antifraude podem ser usados nas transações do provedor de pagamento.
`usesBankInvoiceEnglishName`	Não	`false`	Define se o método de pagamento de fatura bancária usará o nome em inglês (`true`), ou o nome brasileiro (Boleto Bancário)(`false`).
`usesSecureProxy`	Não	`true`	Se `true`, o provedor pode processar o pagamento sem ser certificado pelo PCI . O conector receberá um `secureProxyUrl`em `createPaymentflow`e os dados criptografados do cartão. Se `false`, o provedor deve ser uma entidade certificada pelo PCI e você deve enviar o AOC contendo o fornecido `serviceUrl`.

`requiresDocument`	Não	`false`	Se `true`, o cliente deve incluir o documento do titular do cartão no momento do pagamento. Um novo campo aparecerá no formulário de pagamento. Se `false`, o cliente não precisa incluir um documento do titular do cartão.
`acceptSplitPartialRefund`	Não	`false`	Se `true`, um reembolso parcial será enviado quando ocorrer uma divisão de pagamento. Se `false`, o conector não poderá processar um reembolso parcial quando ocorrer uma divisão de pagamento.
`usesAutoSettleOptions`	Não	`false`	Se ativado `true`, o cliente poderá escolher o comportamento da liquidação automática nas configurações administrativas do VTEX do provedor. As opções disponíveis são: "Usar o comportamento recomendado pelo processador de pagamentos", "Captura automática imediatamente após a autorização do pagamento", "Captura automática imediatamente após a análise antifraude", "Agendado: agenda a captura automática" e "Desativado: não captura automaticamente". Caso contrário `false`, o conector não terá este campo de configuração suspenso para liquidação automática. Mais informações podem ser encontradas no artigo sobre o Recurso de Captura Automática Personalizada .

Solicite uma nova tentativa ao gateway de pagamento.

É necessário realizar uma nova tentativa para desenvolver seu conector de acordo com o protocolo e, para isso, a função abaixo deve ser invocada:

```
this.retry(request)
```

Mais informações sobre o fluxo de novas tentativas podem ser encontradas na seção Autorização de Pagamento do artigo Protocolo do Provedor de Pagamento .

Serviço de provedor de pagamentos

Esta é uma classe que estende o serviço de `@vtex/api`. Você deve invocá-la passando o conector desenvolvido como uma propriedade do primeiro parâmetro, e ela configurará automaticamente as rotas necessárias para você.

O código a seguir mostra como fazer isso e encontrar o item no `node/index.ts`arquivo.

```
import {

PaymentProviderService,

} from '@vtex/payment-provider'



new PaymentProviderService({

connector: YourPaymentConnector,

})
```

Por padrão, o Serviço de Provedor de Pagamentos declara as seguintes rotas:

- `/manifest`
- `/payments`
- `/settlements`
- `/refunds`
- `/cancellations`
- `/inbound`

Se o seu serviço exigir rotas adicionais, você deverá declará-las separadamente e utilizá-las como parâmetros:

```
new PaymentProviderService({

routes: newRoutes,

connector: YourPaymentConnector,

})
```

Se o seu conector exigir clientes adicionais, você também deverá passá-los como parâmetros junto com o conector:

```
new PaymentProviderService({

clients: NewClients,

connector: YourPaymentConnector,

})
```

Utilizando um proxy seguro

Para processar transações com cartões de crédito, débito ou de marca compartilhada, as integrações devem estar em conformidade com os padrões de segurança PCI-DSS . Para integrações hospedadas no VTEX IO que suportam um desses métodos de pagamento, é obrigatório usar o Secure Proxy para fazer chamadas a um endpoint certificado pelo PCI. Você pode conferir mais detalhes no artigo sobre Secure Proxy .

O endpoint deve ser permitido pelo VTEX Secure Proxy. Isso deve ser solicitado por meio de um ticket , enviando o AOC com o endpoint desejado. Atualmente, o Secure Proxy aceita apenas dois tipos de conteúdo: você pode usar `<content-type>` `\"application/json\"`ou `<content-type> `\"application/x-www-form-urlencoded\"`. Qualquer outro tipo de conteúdo não será compatível.

Para fazer chamadas através do nosso Proxy Seguro, você precisa:

1. Estenda a `SecureExternalClient`classe abstrata. O construtor da classe é feito de forma que o VTEX permita `'http://my-pci-certified-domain.com'`que ele seja um dos destinos confiáveis ao receber seu AOC.
2. Defina o URL do proxy seguro na solicitação que você deseja que seja encaminhada pelo proxy. `SecureProxyURL`é recebido no `createPayment`fluxo.

```
import { SecureExternalClient, CardAuthorization } from
'@vtex/payment-provider'

import type {

 InstanceOptions,

 IOContext,

 RequestConfig,

} from '@vtex/api'
```

```
export class MyPCICertifiedClient extends SecureExternalClient {
 constructor(protected context: IOContext, options?: InstanceOptions) {
   super('http://my-pci-certified-domain.com', context, options)
 }


 public myPCIEndpoint = (cardRequest: CardAuthorization) => {
 return this.http.post(
  'my-pci-endpoint',
     {
   holder: cardRequest.holderToken,
   number: cardRequest.numberToken,
    expiration: cardRequest.expiration,
    csc: cardRequest.cscToken
     },
     {
   headers: {
        Authorization: 'my-pci-endpoint-authorization',
       },
   secureProxy: cardRequest.secureProxyUrl,
     } as RequestConfig
   )
 }
```

```
}
```

Fazendo um pedido com seu novo conector

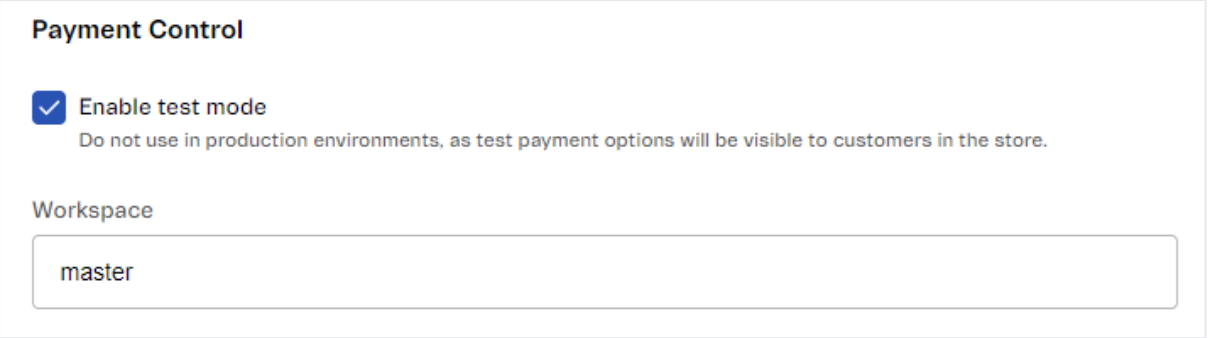
Agora que você tem um novo conector pronto para uso, pode testá-lo completamente no fluxo de produção usando o Checkout da sua loja.

A conta deve ter permissão para testar conectores de E/S. Essa permissão deve ser solicitada por meio de um ticket , informando o nome do aplicativo e a conta onde os testes serão realizados.

Um pré-requisito para este procedimento é ter produtos à venda em sua loja para testes. Para fazer um pedido com seu novo conector:

1. Lance uma versão beta do seu conector. Exemplo: `vtex.payment-provider-test@0.1.0-beta`. Se precisar, consulte o artigo "Como tornar seu aplicativo público" para aprender a criar uma versão beta do seu aplicativo.
2. Instale a versão beta no `master`espaço de trabalho. Aguarde cerca de 1 hora.
3. Acesse `https://{account}.myvtex.com/admin/affiliations/connector/Vtex.PaymentGateway.Connectors.PaymentProvider.PaymentProviderConnector_{connector-name}/`. Substitua `{account}`por o nome da conta que você deseja testar e `{connector-name}`por o nome do seu conector. O formato do nome é: `${vendor}-${appName}-${appMajor}`(ex: `vtex-payment-provider-example-v1`).

4. Em Controle de Pagamentos , ative o ambiente de teste clicando em Ativar modo de teste . Um novo campo chamado Espaço de Trabalho será exibido.
5. Configure o espaço de trabalho como desejar. Você pode deixá-lo como está, como `master`se fosse o espaço de trabalho que você deseja usar para testes.



6. Configure uma condição de pagamento com o conector recém-criado e aguarde 10 minutos para que ela apareça na página de finalização da compra.
7. Faça uma compra utilizando a forma de pagamento que você configurou no seu conector.
8. Após concluir todos os testes de transação na versão beta do conector, publique e implemente uma versão estável do seu conector (ex.: vtex.payment-provider-test@0.1.0). Esta versão estável deve ser enviada para o processo de homologação .

Disponibilizar seu conector para processar vendas.

Para processar vendas com seu conector em todas as contas VTEX, certifique-se de enviar o `billingOptions`campo indicado `free`no manifesto. Se desejar restringir o uso do conector a apenas algumas contas específicas, o `billingOptions`campo também deve ser enviado `free`, mas a equipe de pagamentos deve ser notificada por meio de um ticket de suporte para habilitar o conector somente para as contas desejadas. Mais informações sobre este campo podem ser encontradas no artigo Opções de Faturamento .

O processo de publicação é feito através da loja de aplicativos . Mais informações sobre como fazer isso podem ser encontradas no artigo "Como enviar seu aplicativo para a loja de aplicativos VTEX" .

Em seguida, você precisa abrir um chamado para a equipe de suporte da VTEX informando que a integração foi concluída. No entanto, antes de abrir o chamado, certifique-se de ter as seguintes informações:

- Nome do aplicativo conector : Nome do aplicativo conector PPF. Use o seguinte formato: `"vendor.appname"`. Por exemplo: `partnername.connector-partnername`. Essas informações podem ser encontradas no `manifest.json`arquivo.
- Contato do parceiro : endereço de e-mail do parceiro para o caso de precisarmos comunicar alterações e novos recursos do nosso protocolo.
- Endpoint do provedor de serviços de produção : o caminho base que será usado para chamadas de API ao provedor, por exemplo `https://vtex.pagseguro.com`. Ele deve responder à rota `{{serviceUrl}}/manifest`. Este endpoint deve estar disponível publicamente.
- Contas permitidas : descreva quais contas VTEX deste provedor estarão disponíveis (todas as contas ou contas específicas).
- Novos métodos de pagamento : informa se este conector suporta um método de pagamento que ainda não está disponível no painel de administração do VTEX.
- Fluxo de compra com novo método de pagamento : se um "Novo método de pagamento" for compatível, informe se ele funciona com redirecionamento ou com o aplicativo de pagamento. Para obter mais informações, acesse o artigo Fluxos de compra .

O prazo de SLA (Acordo de Nível de Serviço) exigido para que a equipe de pagamentos realize a homologação é de 30 dias.

Após a conclusão da etapa de homologação, seu aplicativo precisa ser instalado na conta que deseja utilizá-lo. Em seguida, uma nova afiliação estará disponível para configurá-lo.

Atualizando e testando novas configurações para um conector já publicado.

Para alterar e testar novas configurações em um conector já publicado, você deve:

1. Aplique as novas configurações à última versão beta criada do seu conector.

2. Siga os mesmos procedimentos descritos na seção "Como fazer um pedido com seu novo conector" para verificar se a versão beta do seu conector está funcionando corretamente após a aplicação das novas configurações.
3. Crie e envie uma nova versão estável do seu conector (contendo as mesmas modificações da versão beta) para o processo de homologação. Exemplo: vtex.payment-provider-test@0.1.1.
4. Publique-o conforme indicado na seção "Disponibilizando seu conector para processar vendas" .

Implementando um provedor de pagamentos

Um fluxo de compra (também conhecido como fluxo de compras) representa as etapas ou ações que um cliente realiza para concluir o pagamento de uma compra. No VTEX, os fluxos de compra disponíveis são:

- Transparente
- Redirecionar
- Aplicativo de pagamento

Transparente

O fluxo de compra transparente é utilizado quando os clientes preenchem as informações de pagamento no modelo padrão do SmartCheckout e concluem a compra sem sair da loja. Esta é a opção mais comum para quem deseja proporcionar uma experiência de pagamento perfeita.

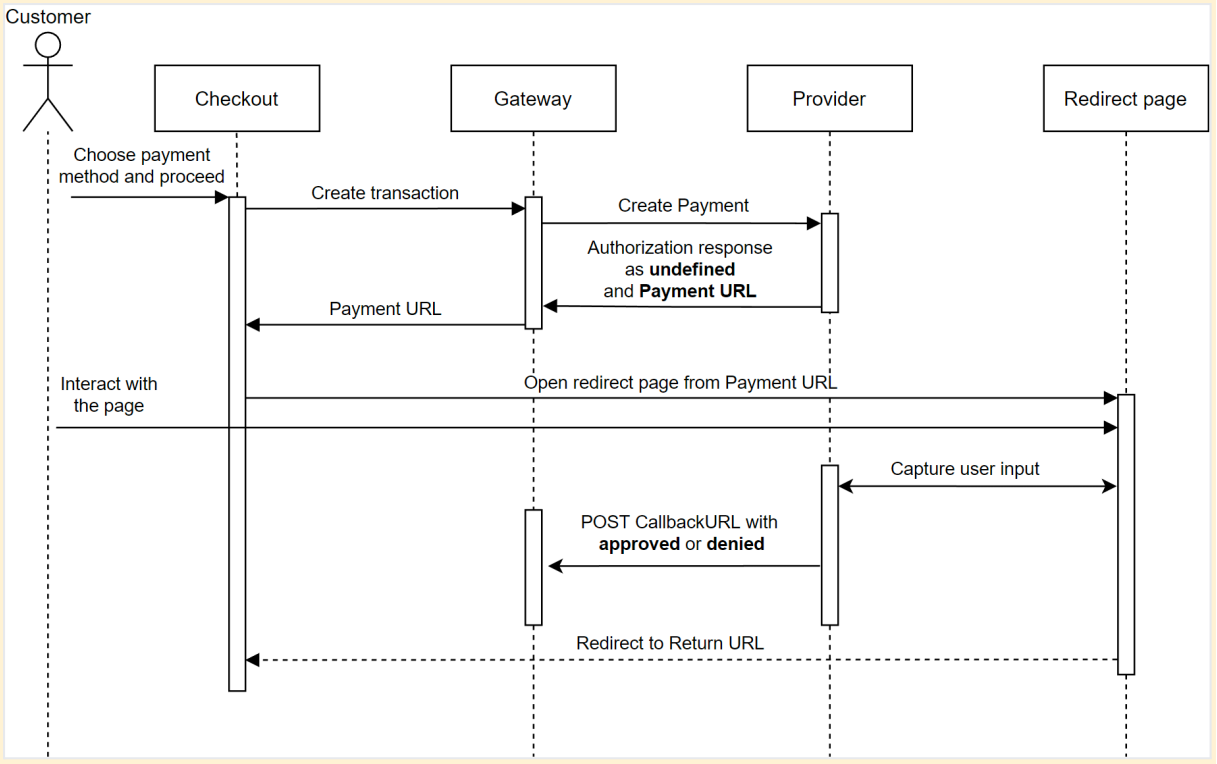
Redirecionar

O fluxo de compra por redirecionamento é utilizado quando os clientes confirmam a opção de pagamento na loja e são redirecionados para uma página diferente para preencher as informações de pagamento e concluir a compra.

Essa opção é utilizada quando o parceiro precisa de uma interação específica com o cliente que não é fornecida pelo modelo padrão do SmartCheckout.

Alguns exemplos incluem: exibir um código QR para pagamento, preencher um formulário de inscrição, fazer login em um site externo.

Para o conector que utilizará o recurso de redirecionamento, não é necessário passar em todos os testes do conjunto de testes, apenas no teste de redirecionamento.



A sequência de eventos em um fluxo de pagamento de redirecionamento ocorre conforme descrito nas etapas a seguir:

1. O cliente clica em "Finalizar Pagamento" no Checkout.
2. O Checkout invoca o Gateway com uma solicitação de Início de Transação .
3. O gateway chama o provedor com uma solicitação de autorização (POST `/payments`).
4. O provedor retorna no corpo da resposta da solicitação de autorização o status `undefined`e o `paymentURL`.
5. O botão de finalização da compra redireciona o cliente para a `paymentURL`página da web.
6. Na `paymentURL`página web, o cliente interage com a interface do fornecedor.
7. O provedor registra as informações da interação com o cliente.
8. O provedor responde com `callbackURL`o status de autorização final (`approved`ou `denied`).
9. O provedor redireciona o cliente para a `returnURL`página da web.

O conteúdo do `paymentURL`parâmetro deve incluir o URL completo para redirecionar corretamente o cliente para a página web desejada, incluindo o caminho do endpoint e um código para esse pagamento específico (ex: `http://php-connector.herokuapp.com/installments.php?paymentId=7ee64e51-a0d3-4405-874c-d7497ab84572`).

Aplicativo de pagamento

O fluxo de compra do aplicativo de pagamento é uma opção flexível que permite aos clientes realizar uma experiência de pagamento personalizada sem sair da loja.

Este modelo se aplica a uma grande variedade de métodos de pagamento graças à sua interação com a API de Checkout. No entanto, os aplicativos de pagamento devem ser desenvolvidos e implementados usando o VTEX IO . Para mais informações, consulte nosso guia de aplicativos de pagamento .

Operações no fluxo de pagamento

As seguintes operações definem os padrões de interação entre o Provedor de Pagamento e o Gateway de Pagamentos VTEX:

- Autorização
- Cancelamento ou reembolso
- Povoado

Quando um cliente conclui uma compra, isso aciona a criação de uma nova solicitação de pagamento, começando com a operação de Autorização.

Autorização

Assim que a compra for concluída, o gateway de pagamentos envia uma solicitação POST de criação de pagamento ao fornecedor.

A carga útil é bastante grande, pois contém informações como os itens no carrinho, endereço de cobrança, ID do pedido, métodos de pagamento, URL de retorno de chamada e outros. Mais informações sobre os endpoints e suas cargas úteis estarão disponíveis no artigo Implementando um Provedor de Pagamento .

Depois disso, o fornecedor deve responder à solicitação. Se o pagamento não puder ser processado imediatamente, o status da transação muda para " `undefined`Em avaliação". Isso não significa que o pagamento foi cancelado ou negado, apenas que está sendo avaliado.

O sistema pode tentar novamente a chamada de forma assíncrona por sete dias até que o pagamento seja efetivamente processado. Durante esse período, a VTEX considerará o pagamento como "pendente".

Para interromper esse ciclo de novas tentativas assíncronas, existem dois fluxos possíveis, dependendo se a integração utiliza ou não a infraestrutura de E/S VTEX:

- Sem VTEX IO (desenvolvido na infraestrutura do parceiro) : quando o provedor recebe um status de pagamento atualizado, ele usa um endpoint de notificação do `callbackURL`parâmetro para alterar o status do pagamento no Gateway para `authorized`ou `denied`. Ambas as respostas encerram o fluxo de autorização.
- Com VTEX IO : quando o provedor recebe um status de pagamento atualizado, ele lê o `callbackURL`parâmetro para chamar um endpoint de nova tentativa . Em seguida, o Gateway faz outra solicitação de Criação de Pagamento para obter o novo status definido pelo provedor, que será `true` `authorized`ou `false` `denied`. Ambas as respostas encerram o fluxo de autorização.

URL de retorno de chamada

O `callbackUrl`campo contém um URL que o provedor de pagamento usa para fazer um retorno de chamada e informar nosso gateway sobre o status final do pagamento: `approved`ou `denied`.

Esta URL possui alguns parâmetros de consulta, incluindo o `X-VTEX-signature`parâmetro `--signature`. Este parâmetro é obrigatório e contém um token de assinatura para identificar que a solicitação foi gerada pela VTEX como medida de segurança. O token de assinatura tem no máximo 32 caracteres. Você pode conferir um exemplo de URL de retorno de chamada com o token de assinatura abaixo:

```
https://gatewayqa.vtexpayments.com.br/api/pvt/payment-provider/transactions/8FB0F11111112222233344449984ACB/payments/A2A9A25B1111111122222233327883C/callback?accountName=teampaymentsintegrations&X-VTEX-signature=R123456789aBcDeFGHij1234567890tk
```

Na página Transações do Admin , o token de assinatura aparece mascarado por motivos de segurança, como neste exemplo: `X-VTEX-signature=Rj******tk`.

Ao fazer a solicitação de retorno de chamada, recomendamos que os provedores de pagamento usem a URL de retorno de chamada exatamente como recebida, o que garante que todos os parâmetros sejam incluídos.

Modo desligado

O status "Modo desativado" é um recurso nativo de confiabilidade que permite à VTEX monitorar a integridade e a estabilidade dos pagamentos de seus parceiros, garantindo que o sistema não cancele pedidos devido a instabilidades.

Quando a VTEX identifica erros consecutivos nas integrações de seus parceiros de pagamento — mais de cinco erros nos últimos cinco minutos — o modo de desativação é ativado para processar transações com cartão de crédito.

Enquanto o modo desativado estiver ativado, todas as transações que permitem um fluxo assíncrono serão mantidas. Durante esse período, a transação permanece em `authorizing`estado de espera.

Esse processo leva em média duas horas e pode armazenar até duas mil solicitações. Se o sistema do provedor voltar à estabilidade, o modo de espera será desativado e nosso sistema tentará processar novamente as transações em espera.

Para obter mais informações e ver as perguntas frequentes, consulte o artigo "Modo Desligado: Perguntas frequentes" em nossa Central de Ajuda.

Cancelamento ou reembolso

O processo de cancelamento é iniciado se o pedido for cancelado por qualquer motivo. Quando um pedido é cancelado, o processo de cancelamento do pagamento também é ativado.

Neste caso, existem dois cenários:

- O pagamento foi aprovado e já foi liquidado.
- O pagamento não foi aprovado.

Pagamento aprovado e liquidado

Quando um pagamento é aprovado e já liquidado, mas cancelado posteriormente, a VTEX realiza a solicitação de reembolso ao provedor de pagamento. Além disso, o Protocolo do Provedor de Pagamento aceita reembolsos parciais quando necessário.

Independentemente da resposta do fornecedor, a VTEX encerrará a transação com o status `canceled`.

Pagamento aprovado, mas ainda não liquidado.

Quando um pagamento é aprovado, mas cancelado antes de ser liquidado, a VTEX faz a chamada POST Cancel Payment para o provedor de pagamento.

Independentemente da resposta do fornecedor, a VTEX encerrará a transação com o status `canceled`.

Pagamento não aprovado

A VTEX envia a solicitação POST Cancel Payment ao provedor de pagamento assim que não houver necessidade de reembolsar um pagamento que não foi liquidado.

Se o provedor não autorizar o cancelamento do pagamento e retornar uma resposta com um `undefined`status, o fluxo será executado de forma assíncrona por um dia. Se o provedor não responder após esse período, o pagamento será cancelado.

Povoado

Assim que o gateway autoriza o pagamento, a VTEX envia a solicitação POST Settle Payment ao provedor. Se o provedor não autorizar a liquidação imediatamente, a solicitação continuará sendo executada de forma assíncrona por um dia.

Durante esse período, quando o provedor responde à chamada autorizando a liquidação, a plataforma da VTEX altera o status da transação `finished`e conclui o fluxo.

Se, após um dia de tentativas, a solicitação falhar, o pagamento será cancelado.

Comunicação no fluxo de pagamento

Ao utilizar a comunicação síncrona, a VTEX se comunicará com o provedor de pagamento durante o fluxo de compra, e isso determinará se o pagamento foi aprovado ou não.

Ao utilizar comunicação assíncrona, a VTEX aguardará um período mais longo pela aprovação do provedor de pagamento. Este é o caso, por exemplo, de boleto bancário, em que o comprador tem alguns dias para efetuar o pagamento do pedido por meio de sua conta bancária.

Dependendo do método de pagamento selecionado, três pontos de extremidade no fluxo de pagamento podem usar comunicação síncrona ou assíncrona:

- POST Criar pagamento
- Cancelar pagamento
- Pagamento pós-venda

Como a escolha do método de pagamento depende do cliente, seu provedor deve estar preparado para receber várias solicitações repetidas nesses pontos de acesso.

Implementando um provedor de pagamentos

Você pode conferir os vídeos de demonstração do aplicativo de pagamento durante nosso horário de atendimento através destes links em inglês e português .

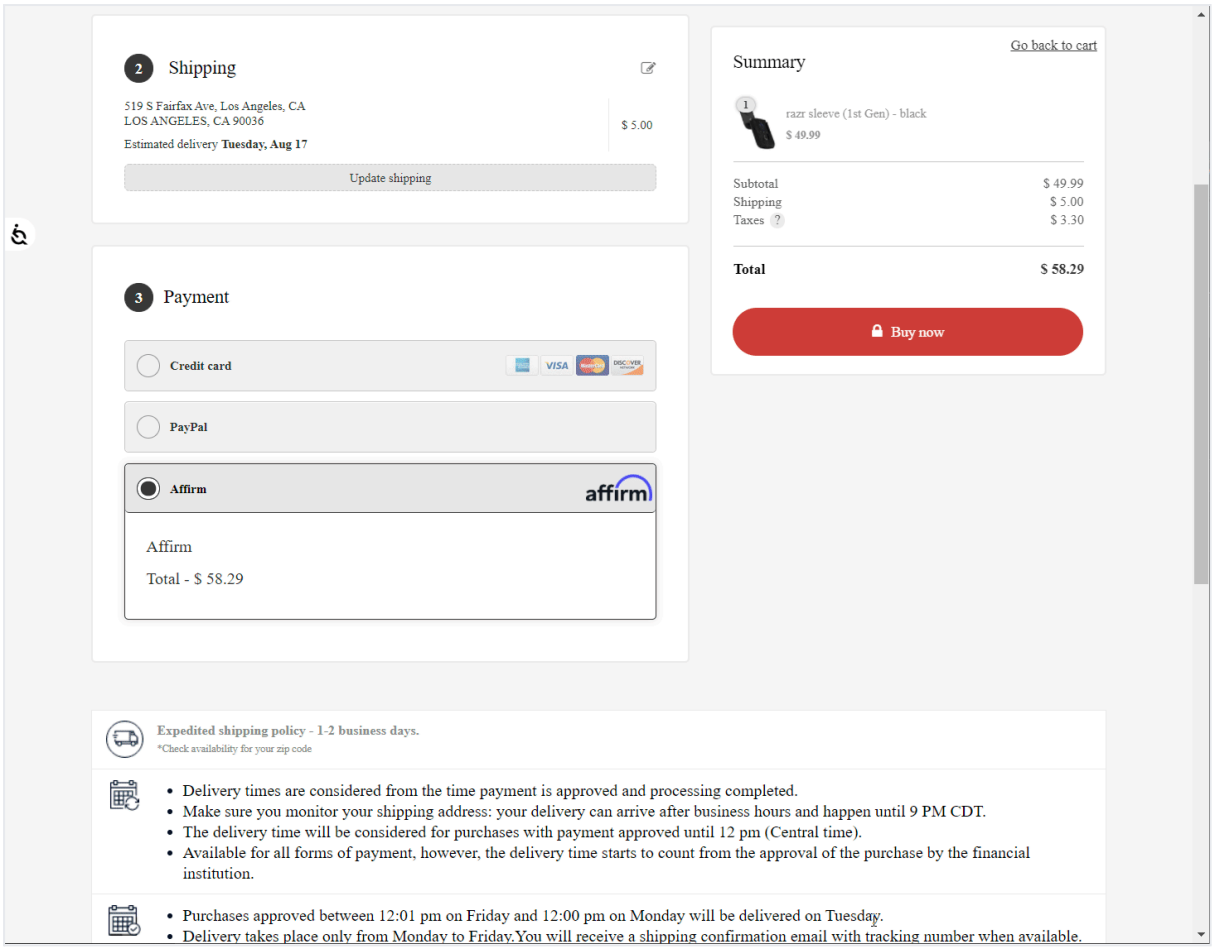
Um aplicativo de pagamento é um tipo especial de aplicativo integrado ao VTEX IO que permite aos desenvolvedores criar experiências de pagamento personalizadas na página de finalização da compra, sem redirecionar os clientes para um site externo.

Em termos de programação, um Aplicativo de Pagamento é definido como uma classe em TypeScript que estende a classe Component do React . Você pode configurar um método de pagamento para usar um Aplicativo de Pagamento específico, e esse aplicativo é instanciado pela interface de finalização da compra quando o método de pagamento é selecionado durante a compra.

Os principais motivos pelos quais alguém optaria por criar um aplicativo de pagamentos em vez de uma integração de pagamentos padrão são as seguintes vantagens:

- Implemente um método de pagamento através do Protocolo de Provedor de Pagamento sem redirecionamento (sem redirecionar os usuários para um site externo).
- Adicione uma camada de segurança aos pagamentos online da sua loja usando o padrão 3D Secure 2 (3DS2) .
- Meça as conversões por meio de ferramentas de análise para fluxos personalizados.

Como você pode ver abaixo, o aplicativo de pagamento é exibido como uma janela modal quando o cliente clica no botão Comprar agora :



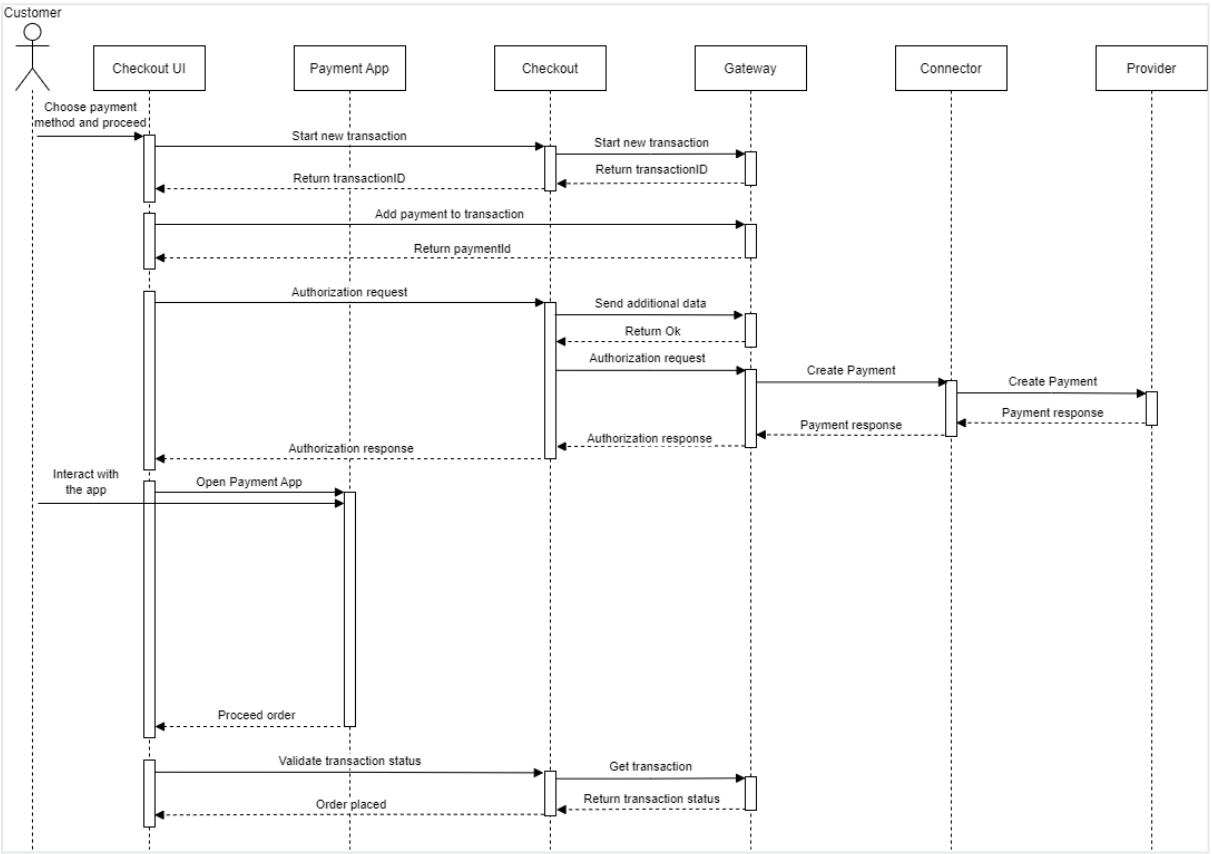
Este tutorial irá guiá-lo pelos passos necessários para desenvolver, testar e configurar um aplicativo de pagamento em sua loja. As seções deste tutorial são:

- Entendendo o fluxo do aplicativo de pagamento : Uma explicação do que é um aplicativo de pagamento, seu comportamento e como usá-lo.
- Implementando um aplicativo de pagamentos : um guia passo a passo com as informações básicas necessárias para desenvolver um aplicativo de pagamentos.
- Modo operacional : Após implantar o aplicativo de pagamento, verifique mais informações sobre a resposta do checkout. Aprenda também como lidar com a carga útil de um pedido e como injetar scripts externos.

Aplicativos de entrada/saída não funcionam em ambientes sem interface gráfica. Se você deseja processar pagamentos nesse tipo de cenário, deve usar uma visualização web de checkout ou implementar uma integração totalmente sem interface gráfica diretamente entre seu sistema e o adquirente/conector, enviando as informações por meio de pagamentos personalizados ou notas promissórias.

Entendendo o fluxo do aplicativo de pagamento

Nosso modelo de aplicativo de pagamento se aplica a uma grande variedade de métodos de pagamento graças à sua interação com a API de Checkout, conforme mostrado no diagrama de sequência a seguir:



A seguir, descrevemos o fluxo na mesma ordem do diagrama de sequência (os passos de 1 a 4 são os mesmos para todos os fluxos de pagamento; os passos exclusivos do fluxo do Aplicativo de Pagamento são os que começam em 5):

1. A interface de finalização de compra envia uma solicitação de início de transação para a API de finalização de compra.
2. A API de Checkout envia uma solicitação de Início de Transação para a API de Gateway.

    - O gateway de pagamento VTEX cria a nova transação e gera um código único `transactionId`, que é retornado ao Checkout e, em seguida, à interface do usuário do Checkout.
3. A interface de finalização de compra envia uma solicitação de envio de pagamentos diretamente para a API do gateway.
    - O Gateway criará, para cada pagamento, uma entidade de pagamento dentro da entidade de transação criada na solicitação anterior. Observe que, para múltiplos pagamentos, todos são enviados nesta mesma solicitação. Ou seja, uma transação pode ter vários pagamentos. Para cada entidade de pagamento, um `Payment` `paymentId`é gerado e todos eles são retornados para a interface do usuário do Checkout. Esta etapa envia as informações de pagamento propriamente ditas (ou seja, os dados do cartão de crédito).
4. A interface de finalização de compra envia uma solicitação de autorização para a API de finalização de compra.
5. A API de Checkout encapsula duas solicitações para a API de Gateway.
    - A primeira é a solicitação de Envio de Dados Adicionais , que envia à API do Gateway informações sobre o pedido, como dados do perfil do cliente, endereço de entrega e informações do carrinho (itens do carrinho). Essas informações são armazenadas com segurança no banco de dados do Gateway e são utilizadas por provedores antifraude, além de alguns provedores de pagamento para autorizar a transação.
    - A segunda é a solicitação de autorização propriamente dita que a API de Checkout realiza para a API do Gateway. Essa solicitação acionará o fluxo de trabalho de Autorização de Pagamento dentro da API do Gateway.
        1. A API do Gateway chama a solicitação Criar Pagamento para o conector correto, dependendo das Condições de Pagamento configuradas pelo administrador da loja.
        2. Neste ponto, o conector processa a solicitação e executa sua lógica dependendo do método de pagamento. Usando o Pix como exemplo, o conector solicitará ao provedor de pagamento que crie um pagamento em sua API e receberá as informações necessárias para gerar o código QR do Pix.
        3. O conector responde à solicitação de Criação de Pagamento para o Gateway. O corpo da resposta possui um `paymentAppData`campo com duas propriedades. A primeira é `vtex_account` `appName`, que descreve o nome da conta VTEX responsável pelo desenvolvimento, manutenção e distribuição do aplicativo, e o nome do Aplicativo de Pagamento que será usado na transação (por exemplo, `vtex_account` `"{vendor}.{appName}"`). A segunda `payload`contém a carga útil para concluir a transação (ou seja, as informações para gerar o código QR Pix). Em seguida, o Gateway responde à API de Checkout, que por sua vez responde à interface

do usuário de Checkout, a qual instancia o Aplicativo de Pagamento com a carga útil correspondente gerada no conector.

6. A interface de finalização de compra instancia o aplicativo de pagamento definido pelo `<PaymentApp>` e adiciona a propriedade `<PaymentApp>` `paymentAppData.appName`ao objeto do aplicativo , preenchendo-a com o conteúdo de `<PaymentApp>`. Em seguida, o aplicativo de pagamento pode executar sua lógica. Por exemplo, pode ser um aplicativo de autorização (3DS v2) ou um código QR (Pix), ou pode simular um redirecionamento dentro do frame do aplicativo de pagamento. Neste ponto, a transação está no estado de autorização (mais informações sobre o status da transação podem ser encontradas no artigo Fluxo de transação em Pagamentos).`appPayloadpaymentAppData.payload`
7. Quando o usuário finaliza a interação, o aplicativo de pagamento é fechado e aciona a validação da transação na interface de finalização da compra. Essa validação inicia o processo, no qual a interface de finalização da compra obtém o status da transação da API de finalização da compra (que, por sua vez, solicita o status da transação da API do gateway). Se o status da transação for "Autorizado" ou "Indefinido" , a interface de finalização da compra exibe a tela de "Pedido Realizado". Caso contrário, a interface de finalização da compra exibe um aviso e retorna ao estado em que o usuário pode escolher o método de pagamento.

Implementando um aplicativo de pagamento

O tutorial a seguir irá guiá-lo pelos passos necessários para implementar um aplicativo de pagamento, incluindo:

- Informações sobre como configurar um ambiente de desenvolvimento.
- Passos para clonar o repositório que contém um código boilerplate de aplicativo de pagamento para você editar.
- Como executar e testar seu aplicativo de pagamento.
- Informações sobre o processo de implantação.

Para prosseguir com os próximos passos, você utilizará a plataforma de desenvolvimento VTEX IO .

Se você estiver implementando um novo método de pagamento em seu aplicativo de pagamentos, entre em contato com a equipe de suporte da VTEX para adicioná-lo ao sistema da VTEX.

Etapa 1: Configurando o ambiente de desenvolvimento

1. Utilizando o terminal do seu computador, instale a CLI (Interface de Linha de Comando) do VTEX IO.
2. Após instalar o VTEX IO CLI, consulte a documentação " Criando um espaço de trabalho de desenvolvimento" para criar um ambiente de trabalho e começar a desenvolver o aplicativo de pagamento.
3. Ao terminar de criar o espaço de trabalho, passe para a próxima seção.

Etapa 2: Desenvolvendo seu aplicativo de pagamento

Estas são as alterações mínimas que você precisa fazer no modelo inicial para criar seu próprio aplicativo.

Além dessas modificações, você precisa adicionar a lógica de negócios do seu sistema de pagamento.

Antes de começar a editar o aplicativo de pagamento, é necessário ter o Git instalado no seu computador. Clique aqui para saber mais.

1. Usando seu terminal, clone o repositório que contém um modelo de aplicativo de pagamento para seus arquivos locais executando o seguinte comando:

```
git clone
https://github.com/vtex-apps/example-payment-authorization-app.git
```

2. Após clonar o repositório, use um editor de código de sua preferência para abrir o projeto localmente.
3. Abra o `manifest.json`arquivo e atualize seu conteúdo de acordo com o seu cenário. Se precisar de mais informações sobre os campos obrigatórios, acesse nossa documentação do Manifesto . O valor definido no `name`campo será o identificador exclusivo deste aplicativo e deve ser o nome exato que corresponde à resposta do seu conector (`appname`propriedade do `paymentAppData`campo no corpo da resposta do endpoint Criar Pagamento). O `billingOptions`campo também é obrigatório e deve ser definido com a propriedade `type`como `free`. Mais informações sobre o `billingOptions`campo podem ser encontradas no artigo Opções de Faturamento .
4. Após concluir a atualização, abra o `pages/pages.json`arquivo e substitua `example-payment-auth-app`o nome do aplicativo `name`definido em `<app_set>` `manifest.json`. O `pages.json`arquivo é responsável por criar as rotas para o seu aplicativo. Portanto, é necessário fazer essa configuração para que o sistema

VTEX encontre e exiba o aplicativo no momento do checkout. Além disso, o `"component"`campo com o `"index"`valor `<app_set>` indica que o `index.js`arquivo na `react`pasta é o arquivo com o componente que será instanciado no momento do checkout.

Substitua apenas a última parte do caminho no `pages.json`arquivo pelo nome do seu aplicativo. Neste caso, a `example-payment-auth-app`parte `<nome_do_aplicativo>`. Não altere o início do caminho `checkout/transactions/`, pois o checkout o utiliza para identificar um aplicativo de pagamento.\n\nAlém disso, se você estiver usando o Test Connector para realizar testes internos com seu aplicativo, você deve manter o nome original do aplicativo que veio do repositório no `pages.json`arquivo, pois ele será a referência para este conector abrir o aplicativo de pagamento no checkout.

5. Usando seu terminal, acesse o diretório do aplicativo e execute o seguinte comando:

```
vtex link
```

Após conectar o aplicativo , os arquivos do seu computador serão sincronizados com nosso ambiente de desenvolvimento na nuvem. Isso significa que qualquer alteração feita localmente no código em que você está trabalhando será enviada para a nuvem e refletida no espaço de trabalho.

Etapa 3: Testando o fluxo de um aplicativo de pagamento

Seguindo os passos abaixo, um aplicativo de pagamento aparecerá na tela de finalização da compra, permitindo que você teste o fluxo geral do aplicativo.

Siga os passos abaixo somente se desejar testar um fluxo do aplicativo de pagamento; caso contrário, pule para o Passo 4: Implantação do aplicativo de pagamento .

1. Certifique-se de que o conector esteja instalado em sua conta, configurado com o aplicativo de pagamento e o método de pagamento que você deseja testar no aplicativo. O conector deve conter o campo `paymentAppData`no corpo da resposta do endpoint Criar Pagamento com as propriedades corretas (`"appName": "{vendor}.{appName}"`e `"payload": "{payload information}"`). Caso ainda não tenha o conector instalado, entre em contato com a equipe de suporte da VTEX para obter ajuda com o processo de instalação. Mais informações sobre como os endpoints devem ser implementados podem ser encontradas no artigo Implementando um Provedor de Pagamento .
2. Selecione a condição de pagamento associada ao seu conector no checkout e finalize a compra para testar o fluxo.

Etapa 4: Implantação do seu aplicativo de pagamento

Após concluir as alterações, siga as instruções da nossa documentação sobre como disponibilizar publicamente a nova versão do seu aplicativo para executá-lo no espaço de trabalho principal .

Modo operacional

Se você seguiu todos os passos anteriores, significa que criou com sucesso um aplicativo de pagamento e está pronto para aproveitar os benefícios de ter uma etapa de verificação adicional no checkout da sua loja.

Esta seção aborda os seguintes tópicos para ajudá-lo a aproveitar ao máximo o aplicativo de pagamentos:

- Manuseio da carga útil do pedido.
- Entendendo a resposta da interface do usuário do Checkout.
- Injetando scripts externos.

Manipulação da carga útil do pedido

Essa `appPayload`é a propriedade que o aplicativo de pagamento recebe e consiste em um JSON serializado com os dados do pedido. Essa variável não é declarada na classe do aplicativo de pagamento, mas é adicionada ao seu arquivo ` `propsPaymentApp.props` após a instanciação do aplicativo na interface de finalização da compra. Você pode acessar essa informação com o seguinte código:

```
const { appPayload } = this.props // This appPayload is a serialized
JSON (as string).
```

O conteúdo esperado é um objeto JSON contendo os dados do aplicativo de pagamento necessários para que ele permita ou negue a realização de um pedido. Portanto, os `appPayload`campos serão definidos pelo desenvolvedor do conector associado.

Para exemplificar como a carga útil pode ser usada, considere o seguinte exemplo:

```
{

   "timeToCancel": 300,

   "transactionApproveLink": "https://..."

}
```

Neste exemplo, existem dois campos para fins de ilustração. O primeiro campo (`"timeToCancel"`) pode representar o tempo que o aplicativo de pagamento permanece aberto antes de cancelar automaticamente a transação. O segundo campo (`"transactionApproveLink"`) pode conter um link para solicitar a aprovação da transação caso todas as condições do aplicativo de pagamento sejam atendidas.

Entendendo a resposta à interface do usuário de finalização da compra.

Quando o aplicativo de pagamento finaliza sua operação, ele precisa informar à interface de finalização da compra que deve ser fechado e, em seguida, verificar se a transação foi aprovada ou não. Se a transação for aprovada `approved`ou não `undefined`, a interface de finalização da compra redirecionará o usuário para a página de pedido realizado; caso contrário, exibirá um aviso e, em seguida, levará o usuário de volta às opções de pagamento.

Para executar essas etapas, a interface do usuário do Checkout fica à escuta do `transactionValidation.vtex`evento disparado pelo aplicativo de pagamento usando o mecanismo nativo de tratamento de eventos do navegador durante a etapa de validação. Quando esse evento é disparado, a interface do usuário do Checkout fecha o aplicativo de pagamento e verifica o status da transação no gateway. O método de exemplo a seguir pode ser usado pelo aplicativo de pagamento para disparar o evento:

```
respondTransaction = () => {

   $(window).trigger('transactionValidation.vtex')

}
```

Se o `transactionValidation.vtex`evento não for acionado, o e-mail de confirmação do pedido não será enviado ao usuário. Recomendamos que você crie um fluxo de repetição para agir nos casos em que esse evento não for acionado ou o e-mail de confirmação do pedido não for enviado.

Injetando scripts externos

A injeção de scripts externos permite que seu aplicativo de pagamento tenha conteúdo e comportamento adicionais. Para executar scripts externos em seu aplicativo de pagamento, você precisa injetar o seguinte script no elemento ` <head>` `head`da página de finalização da compra `html`. Para fazer isso, você precisa realizar uma injeção DOM da seguinte forma:

```
const head = document.getElementsByTagName('head')[0]
```

```
const js = document.createElement('script')

js.id = {{script-id}}

js.src = {{script-src}}

js.async = true;

js.defer = true;

js.onload = {{callback-onload}}




head.appendChild(js)
```

Aqui está um exemplo de injeção de script que insere o Google Recaptcha em um aplicativo de pagamento.

Lembre-se de que, se o `js`script externo manipular o DOM, você deve usar o `Ref` do React para criar um `div`contêiner e passá-lo para a biblioteca. Você pode encontrar um exemplo aqui .

Cenários de Aplicativos de Pagamento

Nas seções seguintes, você encontrará alguns cenários em que poderá usar um aplicativo de pagamento.

Cenário 1: Uma integração via Protocolo de Provedor de Pagamento que requer um novo método de pagamento.

A melhor maneira de criar um novo método de pagamento é desenvolvendo um aplicativo de pagamento, pois ele permite que os clientes concluam a transação sem sair da página de finalização da compra. Além disso, permite adicionar uma camada de segurança aos pagamentos online da sua loja, utilizando os padrões do 3D Secure 2 .

Cenário 2: Aplicativo de pagamento e 3D Secure 2

O 3D Secure 2 é um protocolo que permite que o processo de finalização de compra de um site esteja em conformidade com os requisitos de Autenticação Forte do Cliente

(SCA, na sigla em inglês) , realizando uma autenticação baseada em risco para aprovar uma transação online com cartão. Ele foi criado para atender às regulamentações da Diretiva de Serviços de Pagamento 2 (PSD2, na sigla em inglês) revisada na Europa.

O 3D Secure 2 (3DS2) é uma evolução do 3D Secure 1 (3DS1). Ele introduz um fluxo de autenticação sem atritos e aprimora a experiência de compra em comparação com o 3DS1. Mais detalhes sobre as diferenças entre o 3DS1 e o 3DS2 podem ser encontrados nas Perguntas Frequentes do 3D Secure 2 .

Na VTEX, o 3D Secure 2 só pode ser implementado através do aplicativo de pagamento, pois a plataforma não suporta métodos alternativos, como URLs de redirecionamento para esse protocolo.

Fluxo 3DS2

Para aplicar o 3DS2 ao processo de finalização da compra, o adquirente que processa os pagamentos do site precisa criar um aplicativo de pagamento que lide com os desafios de cada banco emissor que utilizará a autenticação forte por meio do 3DS2.

Quando o cliente adiciona as informações do cartão de crédito e clica no botão Finalizar Compra, o adquirente entrará em contato com o banco emissor através do fluxo 3DS2.

A emissora do cartão, por sua vez, analisará o contexto de fraude daquela compra (a pontuação, probabilidade), também chamada de Autenticação Baseada em Risco . A pontuação será o resultado de uma série de elementos baseados em risco, que podem incluir o valor da compra, o histórico de transações, informações do dispositivo, entre outros. Se essa análise resultar em uma pontuação de alto risco de fraude, o provedor poderá optar por solicitar o desafio 3DS2, pedindo ao cliente que prossiga com o método de Autenticação Forte (ou seja, aprove o pagamento no aplicativo do banco). Se a pontuação indicar baixo risco de fraude, o provedor não solicitará o desafio.

Caso o adquirente decida que deve realizar a Autenticação Forte para qualquer pagamento, ele deve retornar à VTEX, no corpo da resposta da rota Criar Pagamento , suas informações `status`de usuário `undefined`e de aplicativo de pagamento no `paymentAppData`campo indicado.

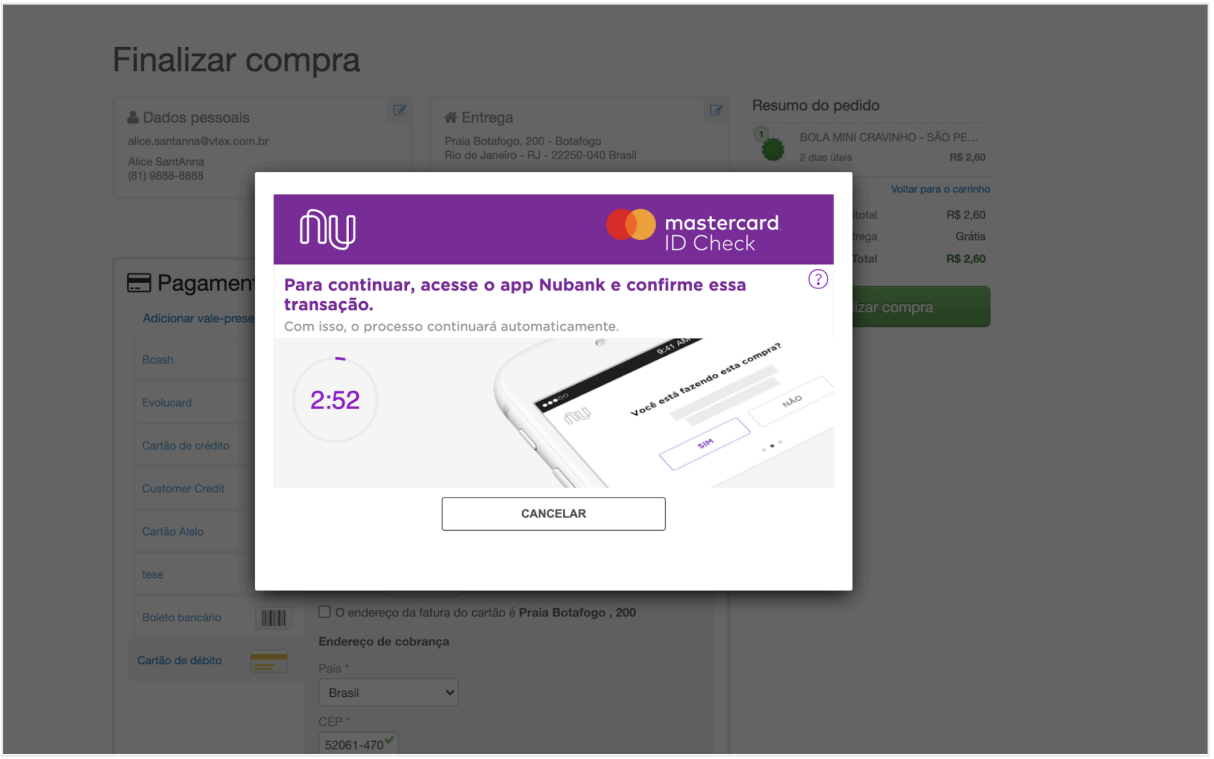
O `status`código retornado `undefined`indica ao nosso gateway que o pagamento ainda não foi autorizado nem negado. Isso pode ocorrer por diversos motivos, sendo a autenticação forte um deles. O que determinará se o aplicativo de pagamento, que utiliza o protocolo 3DS2, será acionado é o `paymentAppData`campo presente na resposta.

O Gateway salvará os dados deste aplicativo no pagamento e informará ao Checkout que o pagamento ainda não foi autorizado nem negado. Isso fará com que a interface do Checkout execute o aplicativo de pagamento especificado no `paymentAppData`campo.

Assim que o aplicativo de pagamento for aberto, o aviso de Autenticação Forte será exibido ao cliente na tela de finalização da compra. Será necessário que o cliente aprove a transação em sua conta no banco emissor (ou seja, no aplicativo do banco).

Após finalizar a interação com o aplicativo de pagamento, o aplicativo é fechado e redireciona o cliente dependendo do status `status`da transação. Se o status for "Em andamento" `approved`ou ainda estiver em andamento `undefined`, o usuário será direcionado para a tela de confirmação do pedido. Se o status for "Em andamento" `denied`ou " `canceled`Em espera", a interface de finalização da compra exibirá um aviso e retornará ao estado em que o usuário pode escolher o método de pagamento.

A imagem a seguir mostra como o 3DS2 é apresentado aos clientes:



Implementando um provedor de pagamentos

Os métodos de pagamento são diferentes maneiras pelas quais um cliente pode usar dinheiro para comprar um produto ou serviço. Na VTEX, os tipos de métodos de pagamento disponíveis são:

- Cartões de crédito
- Cartões de débito
- Dinheiro
- Pagamentos personalizados
    - Promissória
    - Marcas Próprias
    - Co-branded
- Pagamentos regionais
    - Pagamentos instantâneos (Pix/Brasil)
    - Fatura Bancária (Boleto Bancário/Brasil)

Diferentes países utilizam diferentes métodos de pagamento. Existe uma extensa lista de métodos de pagamento em todo o mundo, especialmente porque eles variam de país para país. Antes de integrar com a plataforma VTEX, é importante verificar se os métodos de pagamento processados pelo seu sistema são compatíveis com os nossos.

Cartões de crédito

A maioria das instituições financeiras oferece um limite de crédito aos seus clientes. Esse crédito pode ser usado como forma de pagamento por meio de um cartão de crédito. Dessa forma, o cliente pode fazer uma compra e pagá-la em um futuro próximo.

Além disso, há mais um aspecto importante sobre cartões de crédito: as bandeiras corporativas — as redes responsáveis pela gestão das operações com cartões de crédito. MasterCard, Visa, Amex e Diners são exemplos de redes de cartões de crédito no setor de pagamentos. Para mais informações, consulte Fluxo de pagamento com cartão de crédito.

Cartões de débito

Um cartão oferecido por uma instituição financeira aos seus clientes. Ao usar o cartão de débito como forma de pagamento, as compras são pagas imediatamente, com o valor sendo debitado diretamente da conta bancária do cliente.

Assim como os cartões de crédito, os cartões de débito também funcionam em conjunto com bandeiras corporativas.

Dinheiro

Os pagamentos em dinheiro podem ser ativados no aplicativo inStore. Após configurar e ativar esse método de pagamento, o pagamento em dinheiro poderá ser recebido na loja física ou no momento da entrega do produto ao consumidor, em caso de serviço de entrega. Para mais informações, consulte a seção " Configurando o pagamento em dinheiro pelo inStore".

Pagamentos personalizados

Existem alguns métodos de pagamento que não seguem os padrões do mercado de pagamentos — os Pagamentos Personalizados. Isso significa que esses métodos de pagamento têm um comportamento muito específico para cada cenário em que podem ser aplicados.

Na VTEX, trabalhamos principalmente com três tipos: Promissória, Marca Própria e Marca em Conjunto.

Promissória

O vendedor precisa aprovar manualmente cada pagamento calculado em nosso sistema. Após a aprovação, a transação prossegue normalmente. A nota promissória é utilizada principalmente para facilitar pagamentos em dinheiro.

Marcas Próprias

É um cartão de crédito feito exclusivamente para uma loja que opera com marca própria.

Co-branded

Além disso, um cartão de crédito feito exclusivamente para uma loja. A diferença entre uma marca própria e uma marca compartilhada é que a marca compartilhada funciona por meio de uma parceria entre uma bandeira de cartão de crédito corporativa — como MasterCard e Visa — e a marca da loja.

Pagamentos regionais

Pagamentos instantâneos (Pix/Brasil)

O Pix é o ecossistema de pagamentos instantâneos implementado pelo Banco Central do Brasil (BCB) para viabilizar transferências de dinheiro online com custos reduzidos, maior segurança e disponibilidade 24 horas por dia, 7 dias por semana. As transferências ocorrem diretamente da conta do pagador para a conta do beneficiário, sem a necessidade de intermediários, resultando em custos de transação menores. Para mais informações, consulte Pix: Pagamentos Instantâneos no Brasil .

Fatura Bancária (Boleto Bancário/Brasil)

O boleto bancário é um método de pagamento popular no Brasil. Consiste em um comprovante oficial que o cliente pode pagar em dinheiro em mais de 200 mil locais de pagamento, como bancos, correios e supermercados. Também pode ser pago eletronicamente por meio do internet banking. Para mais informações, consulte o fluxo de pagamento do boleto bancário .

Embora existam facilidades de pagamento, o desembolso pode levar até dois dias úteis para ser processado.

Implementando um provedor de pagamentos

A última etapa para concluir o processo de implementação é verificar se a integração foi feita corretamente.

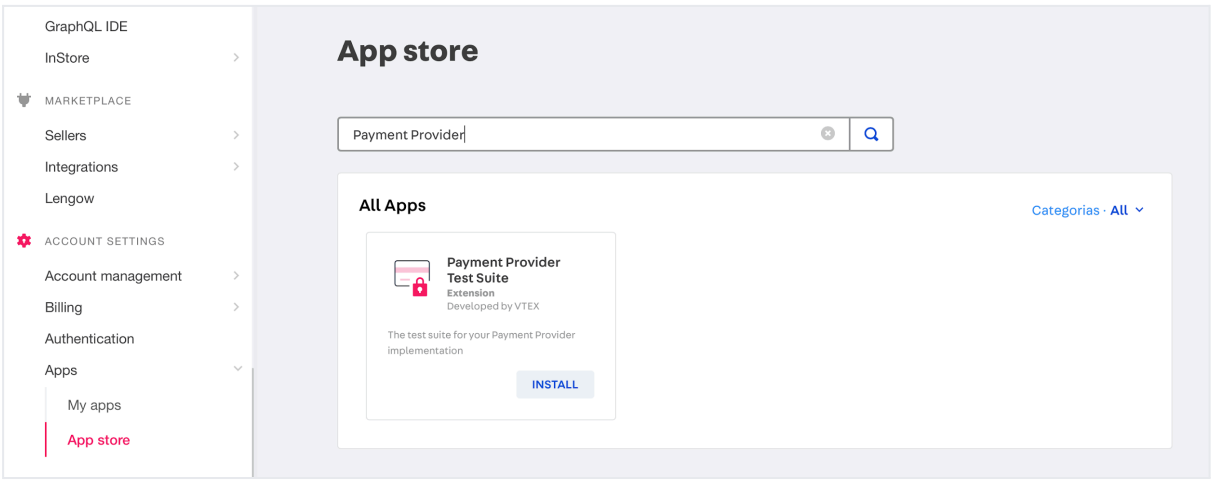
Para isso, você pode simular o funcionamento da integração com o aplicativo Payment Provider Test Suite.

Instale o aplicativo Test Suite.

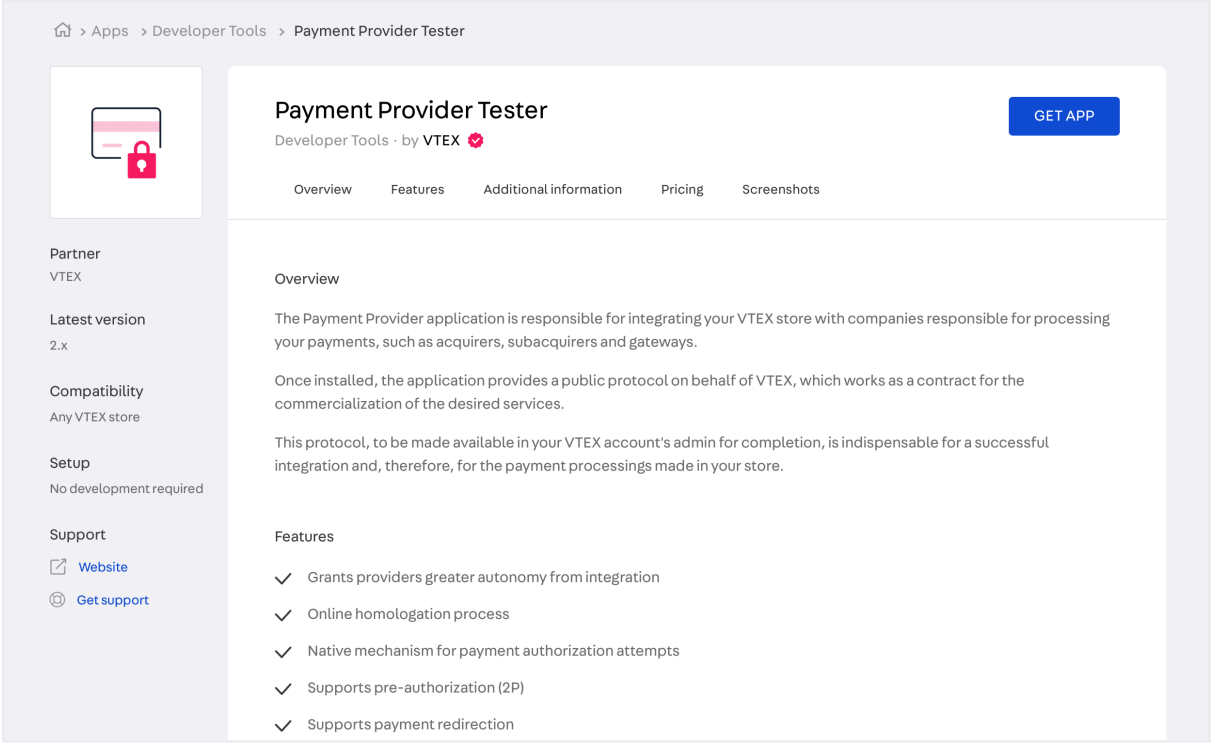
Existem duas maneiras de fazer o download: acessando a Loja de Aplicativos VTEX ou através do painel de administração.

Considerando que você deseja baixar o aplicativo pelo painel de administração, proceda da seguinte forma:

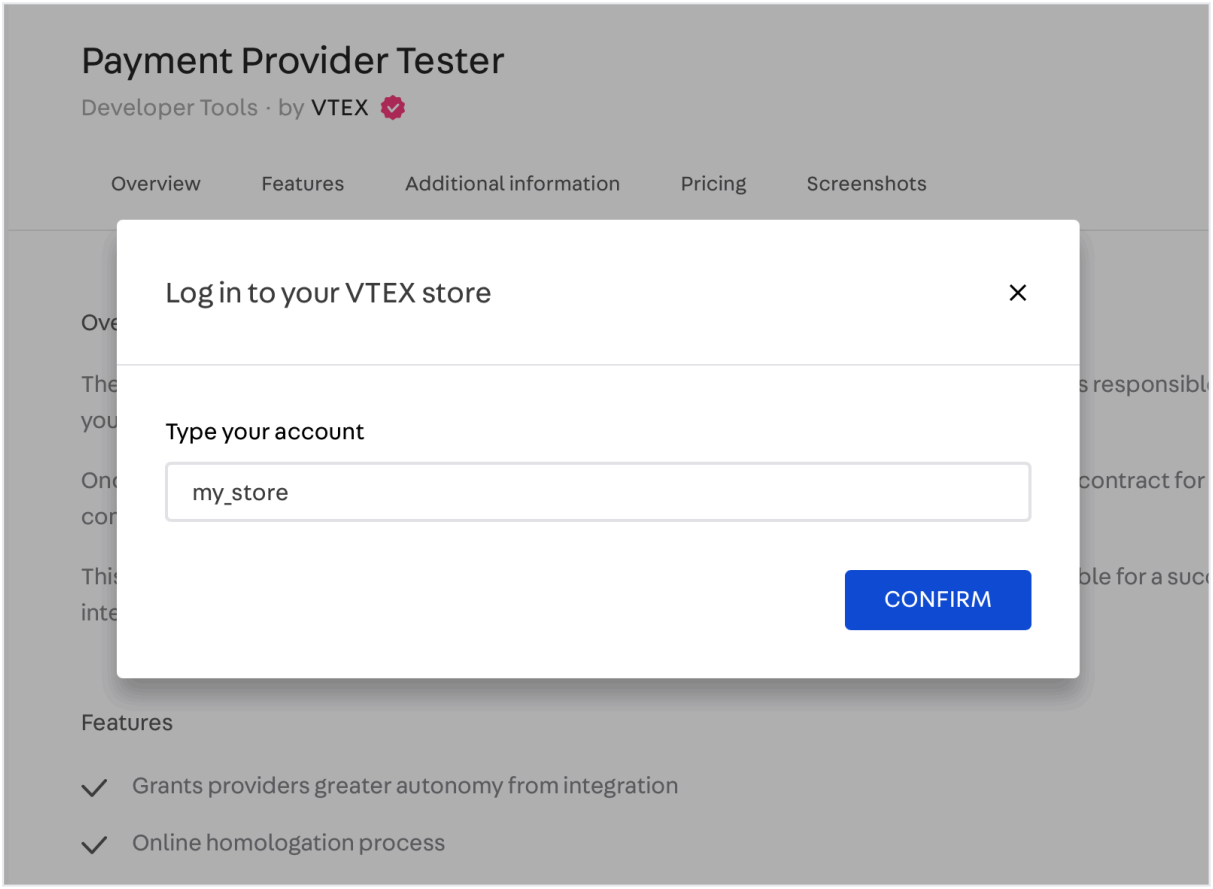
1. Acesse o painel de administração.
2. Acesse o módulo CONFIGURAÇÕES DA CONTA .
3. Clique em Aplicativos e depois em App Store .
4. Agora procure por "Payment Provider Test Suite" na barra de pesquisa.
5. Clique em Instalar .



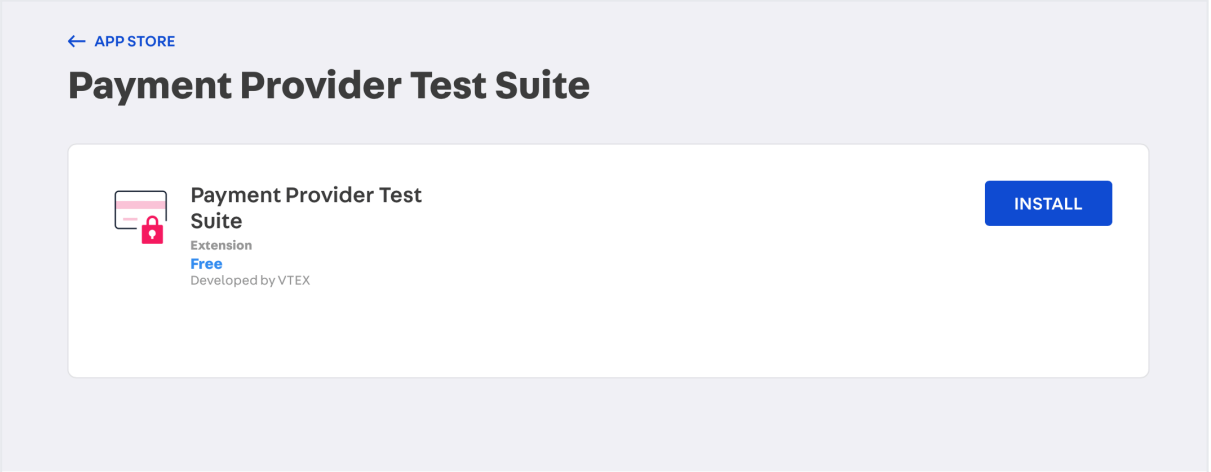
6. Em seguida, você será redirecionado para a loja de aplicativos VTEX. Clique no botão OBTER APLICATIVO no canto superior direito da página.



7. Em seguida, na janela pop-up, digite o nome da sua conta - em letras minúsculas e sem espaço entre as letras - e clique no botão CONFIRMAR .



8. Clique em INSTALAR para concluir o processo de instalação.



Depois disso, o aplicativo Test Suite será instalado em seu painel de administração.

Execute os testes

Após concluir o processo de homologação, você pode começar a executar testes para validar a integração.

Informações sobre o serviço

Para começar, preencha cada campo de acordo com as instruções abaixo:

- URL do serviço: O endpoint do provedor. A VTEX usará esta URL para contatar o sistema do provedor.
- Nome do Conector: Como seu conector será nomeado no VTEX. O nome do conector deve estar necessariamente relacionado à sua marca. Além disso, o nome do conector diferencia maiúsculas de minúsculas. Isso significa que considera letras maiúsculas e minúsculas.
- X-VTEX-API-AppKey: preencha com o valor `X-VTEX-API-AppKey`.
- X-VTEX-API-AppToken: preencha com o valor `X-VTEX-API-AppToken`.

Uma vez definido o nome do conector, ele não poderá ser alterado.

Em seguida, clique no botão Verificar URL . Essa ação chamará o endpoint GET List Payment Provider Manifest . A resposta indicará quais métodos de pagamento poderão ser analisados na próxima etapa (formulário de testes).

Testes

Após a chamada, os métodos de pagamento implementados serão exibidos no formulário e as opções serão ativadas automaticamente.

Verifique se as opções corretas foram ativadas. Em seguida, clique no botão Executar testes .

Registros

Todas as operações realizadas na fase de teste serão expostas na caixa de Logs, incluindo aquelas que possam apresentar algum erro na integração.

Caso ocorram erros, faça todos os ajustes necessários para adaptar seu conector às regras do Protocolo do Provedor de Pagamentos e tente executar o teste novamente.

Se tudo estiver correto, você precisa abrir um chamado para a equipe de suporte da VTEX informando que a integração foi concluída. No entanto, antes de abrir o chamado, certifique-se de ter as seguintes informações:

- Nome do conector : uma descrição do provedor. Use no máximo 16 caracteres alfanuméricos. Este nome não poderá ser modificado após a publicação.
- Contato do parceiro : endereço de e-mail do parceiro para o caso de precisarmos comunicar alterações e novos recursos do nosso protocolo.
- Endpoint do provedor de serviços de produção : o caminho base que será usado para chamadas de API ao provedor, por exemplo. `https://vtex.pagseguro.com>`Ele deve responder à rota `{{serviceUrl}}/manifest`. Este endpoint deve estar disponível publicamente.
- Endpoint do provedor de serviço Sandbox : o caminho base que será usado no modo de teste para chamadas de API ao provedor. Exemplo `https://sandboxserviceproviderendpoint.com`:
- Conta do proprietário : o nome da conta VTEX que será usado nas solicitações de retorno de chamada. Esta conta deve estar disponível em *[conta].myvtex.com* .
- Contas permitidas : descreva quais contas VTEX deste provedor estarão disponíveis (todas as contas ou contas específicas).
- Novos métodos de pagamento : informa se este conector suporta um método de pagamento que ainda não está disponível no painel de administração do VTEX.
- Fluxo de compra com novo método de pagamento : se um "Novo método de pagamento" for compatível, informe se ele funciona com redirecionamento ou com o aplicativo de pagamento. Para obter mais informações, acesse Fluxos de Compra .

O prazo de SLA (Acordo de Nível de Serviço) exigido para que a equipe de pagamentos realize a homologação é de 30 dias. Este prazo começará a contar somente após o envio do documento Acordo Mestre de Parceria para Serviços Financeiros (MPA) . Para solicitações de homologação sem a utilização do MPA, o prazo de SLA poderá ser

estendido devido à necessidade de análises adicionais por parte da equipe de pagamentos.

Não abra o chamado para homologação se a caixa de Logs apresentar qualquer tipo de erro. Refaça a operação até que os resultados na caixa de Logs estejam corretos. Se precisar de ajuda nessa situação, entre em contato com seu Gerente de Contas de Parceiros.

É obrigatório abrir um chamado com o Suporte da VTEX. Sem isso, a implementação não será reconhecida pelo nosso sistema.

Em seguida, a equipe de suporte da VTEX informará se a implementação foi realizada corretamente.

Quando a homologação do provedor de pagamentos não é necessária?

Um conector só fica isento do processo de homologação do provedor de pagamentos se todas as condições abaixo forem atendidas simultaneamente:

- É um conector PPF , desenvolvido usando VTEX IO .
- Utiliza apenas métodos de pagamento já disponíveis na plataforma VTEX.
- É instalado localmente e restrito a contas específicas.
- A conta de destino já está utilizando algum conector IO/PPF.

Caso alguma dessas condições não seja satisfeita, o conector deverá passar pelo processo de homologação.