Payment Extensibility Starter Kit

This guide explains the steps needed to create and deploy a new payment gateway. Contact your Kibo enablement team before setting up a new payment gateway to discuss your needs and plan for the resting and certification process.

If you want to create your own tax integration (such as if you want to use a tax service other than Avalara), you can use an API Extension or an application capability. See Creating a New Tax Integration for more information.

Create a Dev Center Payment Gateway Application

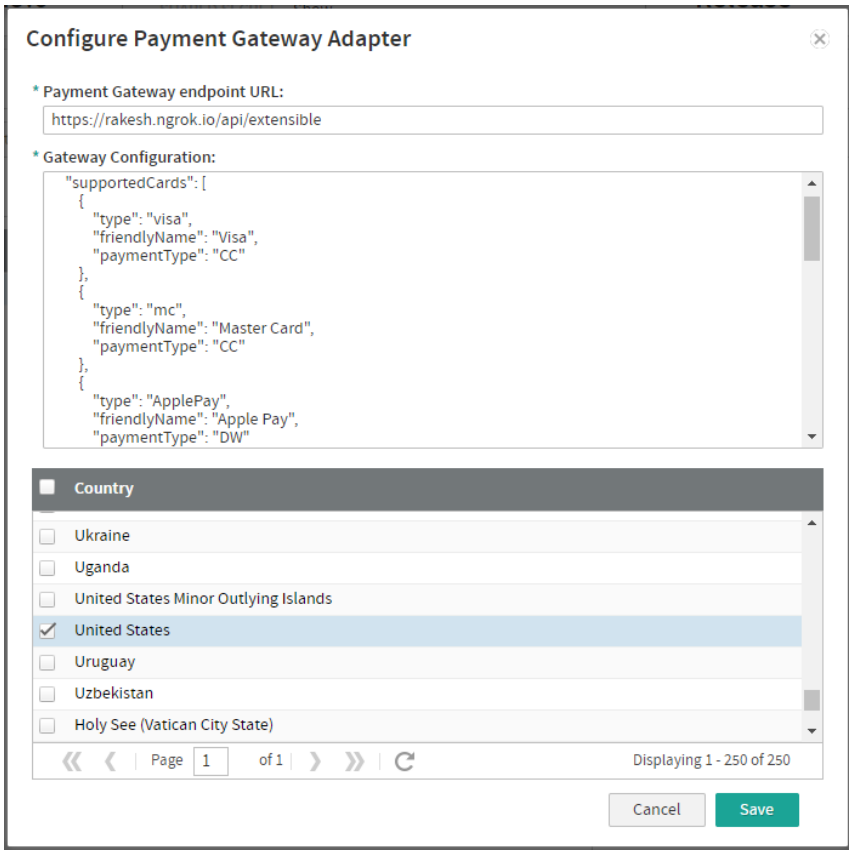
An application must be created first to access the environments.

1. Log into Dev Center.
2. Click **Develop** > **Applications**.
3. Click **Create Application**.
4. Enter in a name.
5. Click **Save**.

Configure Metadata for your Gateway

You must select the gateway adapter capability for your new application and configure its metadata.

1. On the left hand side, select **Packages** > **Capabilities**.
2. Select **Add Capability**.
3. Select **Payment Gateway Adapter** from the dropdown menu.
4. Click **Ok**. This displays the Configure Payment Gateway Adapter screen.



5. In the Gateway Configuration field, a template is displayed for you to enter configurations for your payment gateway in JSON format. If you are creating a payment gateway that processes gift cards, you must add the `supportsGiftcard` and `schemaVersion` fields to your gateway configuration after the administrationUi section as in the following example:

```
"administrationUi": [
//administration UI configs
],
"supportsGiftcard" : true,
"schemaVersion" : "1.0"
}
```

   The `supportsGiftcard` field must be set to true, and the `schemaVersion` field must be set to 1.0.

6. Depending on what kind of payments your connector can process, you need to set the following configurations.
   - Set `paymentType` field (`supportedCards.paymentType`) to one of the below values based on payment type.
     - CreditCards: "CC"
     - DigitalWallets: "DW"
     - ThirdPartyPayments:"3P"
   - Populate the features collection with at least one of these values, such as in the example below.

        - "CreditCards"
        - "DigitalWallets"
        - "ThirdPartyPayments"

          ```
          "features":[
          "CreditCards",
          "DigitalWallets"
          ],
          ```

    - When setting administration UI fields for the connector, use the `xtype` field to configure the type of field you want to display in the admin.
        - "textfield"
        - "password"
        - "dropdown"
        - "radio"
        - "email"
        - "date"
        - "checkbox"
        - "number"
7. Select the countries whose payments you want this payment gateway to process.

Fork/Clone the Payment Gateway Template

Kibo provides a template repository to help you configure your new gateway.

1. Create a repository by either forking or cloning the Payment Gateway Template.
2. Implement the adapter.
3. Update the included tests.

Implement Custom Endpoints

Your custom application needs to include an endpoint that allows your storefront theme to call the adapter directly. This will start a session with the gateway outside of the flow of an out-of-the-box credit card payment.

For example, the below endpoint will start a session with Apple Pay. The `methodName` field is important, because if the payment gateway adapter handles responses at /session then the Kibo Composable Commerce Platform will pass the request to this endpoint and run your custom code.

```
self.applePayToken.apiModel.thirdPartyPaymentExecute({
  methodName: "Session",
  cardType: "ApplePay",
  body: {
    domain: window.location.hostname,
    storeName: self.storeName,
    validationURL: validationURL
  }
})
```

If a token is saved, such as for a Google Pay implementation, you can also pass in the ID that you receive from that response. The payment service will get the full token from Kibo's database and pass it to the gateway adapter, which will then have access to the payment data saved in the token.

```
token.apiThirdPartyPaymentExecute({
  methodName: "session",
  cardType: "GooglePay",
  tokenId: "<token id returned from apiCreate>",
  body: {
     domain: window.location.hostname,
     storeName: "test",
     validationURL: "some url"
  } }).then(console.log);
```

Extensibility allows you to create any amount of endpoints. If a digital wallet requires two separate endpoints to achieve its logic, both of those can be added to the adapter and will be accessible from the storefront.

You can also view all default gateway endpoints at this repository. Any custom gateway can have a version of each function listed there, so you can use this repository as a template for configuring your gateway.

Gateway Adapter Hosting

While in development, Kibo allows adapter code to be hosted either remotely or inside of Kibo infrastructure. Hosting the adapter yourself may be advantageous if you require access to logs. For certification or "production" deployment, Kibo must host the code to fulfill a set of standards and complete the audits mentioned below.

Self Hosting an Endpoint

Follow the below steps if you plan on hosting the adapter yourself.

1. In Dev Center, open the application details page for your payment gateway application.
2. Click **Packages**, and then click the **Details** tab.
3. Under **Configuration URL**, add the URL endpoint of the payment gateway application’s configuration page.
4. Save the application.

Kibo Managed Endpoint

Follow the below steps when you want Kibo to host the adapter for use on a production tenant.

1. Create a 'dev' branch in your repo (this will be the branch that will be deployed).
2. Provide git repo access to your Kibo enablement team.
3. The representative will provide an ETA.
4. You will then be contacted when a hosted development version of the adapter is available for a sandbox.
5. Test the sandbox and sign off on the adapter.

Prepare the Gateway for Production Certification

After you have adequately integration tested your gateway, submit it for certification so that it can be used by a production tenant. Before doing so, validate the following:

- Your gateway is built using the latest version of the template and references the latest "kibo-paymentgateway-hosting" package.
- All tests pass.
- No linter errors are present.
- Code coverage is greater than 80%.
- Any endpoints that the your code uses are either hard coded or managed by static Production/Sandbox settings.
    - Note that endpoints cannot be passed in through admin user configuration.
- No PCI data is logged.
- Logging is limited to discrete items, not complex objects.

Submit the Gateway for Certification

To certify the application:

1. Provide your Kibo enablement team with the Application ID and Development account of the gateway application. They may also want you to include the following additional information:
    - Git version tag
    - GIT-SHA
    - List of external endpoints that the code calls.
2. Kibo will perform a code review and PCI audit. This usually takes about four weeks.
3. If approved, you will be notified that the certified gateway adapter can be installed on a production tenant.