Icon

Description automatically generated

Instore Integration  
Partner Certification

Introduction

In order to facilitate a payment workflow on the Zip NZ platform, merchant point of sale systems must integrate with the Zip NZ API in order to initiate the workflow. This document describes a typical integration workflow.

All requests to the Zip NZ API are to be made from the Merchant Server (i.e. server to server communication).

# API Documentation

Documentation for the Zip NZ API can be found at https://docs-nz.zip.co/instore-api.

# Authentication

All calls to the Zip NZ API need to be authenticated. Zip NZ uses OAuth (Client Credentials Flow) to authorise requests to the API.

# Payment Flow

The below diagram shows the flow between the merchant POS and Zip NZ API for a typical payment process

Diagram

Description automatically generated

Each of these steps is described in more detail below:

1. **Operator selects Zip NZ as payment method**  
   Once all items have been scanned, Zip should be presented as a payment method
2. **Operator inputs customer’s Zip Instore code into POS**Once the customer logs in through the Zip app/portal to generate a 6-digit/QR instore code, the operator should then input this code into their POS
3. **Merchant Server POSTs to Zip NZ API**  
   The merchant server then creates an API call to the Zip NZ ‘Create POS Order’ endpoint. This contains:
   * Customer In-Store Code (6-digit/QR pre-approval code)
   * Operator Name
   * Order details (Item name, quantity, price etc.)
   * Amount (Total amount to be charged to the customer)
   * Merchant Reference
4. **Zip App/Portal requests confirmation of order from the customer**Once the order is created, the customer’s Zip App/Portal will update and ask for approval of the order. The POS should show the payment as pending/awaiting confirmation with the option to cancel if needed
5. **Merchant Server Polls Zip NZ API for Order Status**The merchant server should poll Zip to enquire whether the order has been approved or cancelled. The status returned can be the following:
   * Pending: the order has been created, awaiting user response
   * Approved: the order has been approved and funds held by the customer. This status is only relevant to orders created with the auth flow
   * Complete: the order has been paid
   * Declined: the order was declined by the customer
   * Expired: the order has timed out
   * Cancelled: the order has been either cancelled by the merchant prior to the customer approving/declining, or the order has been rolled back by the merchant. This status is only relevant to orders created with the auth flow
6. **User approves the Zip order, and the POS shows a successful order**  
   Once the user has approved the Zip order and the order status is complete, they are then shown a success page on their Zip App/Portal. For the operator, their POS should reflect this

# Refund Flow

In the event of a refund, Zip NZ provides an endpoint to initiate the refund of monies to the customer.

The refund API endpoint can be called with a (Zip returned) orderId and the unique merchant refund reference. It is also designed to be idempotent, in the event of any transient failure.

# Scopes

**Merchant Level**

For larger retailers who wish to use an integration/middleware layer, we can setup your environment so there is only one Auth client. This means all auth-related concerns can be handled within your middleware layer and originating store/terminal details are pass through the subsequent API calls.

In these cases, you will need to stipulate the storeId in the relevant API calls via a header.

**Store Level**

We can also setup an auth client per-store, which means each store has a separate auth client. This is useful for cases where a retailer does not run a middleware layer. We suggest scoping the auth credentials to the individual stores in this case, as any compromised keys are easier to invalidate & re-issue and do not risk the whole retail group.

**Separate Settlement**

Please note that at this time, we equate a merchant as the payment relationship, meaning if each of your stores have a separate settlement bank account, the store level scope cannot be utilised. In this case, separate merchant credentials will need to be allocated for each store.

# Acceptance Criteria

### Payment Method at Checkout

* The operator is able to select Zip NZ as a Payment Method at checkout

### Order Creation

* Upon entering the Zip Instore code into the POS, an order will be created
* Once the customer proceeds and approves the order, they should be presented with a success message
* The merchantReference for the created order should match the Order Number/Reference on the POS platform

### Order Confirmation

* When the operator has successfully completed the Zip checkout, they should be presented with an order confirmation message on their POS

### Order Abandoned

* When a operator manually exits from the Zip checkout without completing the transaction, the order should promptly be cancelled and the operator should be redirected back to their POS to select a different payment method
* When the customer chooses to cancel the order, the order should then be promptly cancelled and the operator should be redirected back to their POS to select a different payment method

### Refund

* The operator should be able to refund any successful Zip order - either for the full amount or a partial amount of the order

# Checklist

## Overview

Each of the items in the below document should be completed for a successful integration. If any items remain unclear, please reach out to our team and refer to the relevant sections of the API docs where all these items are outlined:

## 

## Test Results

### Onboarding

|  |  |
| --- | --- |
| **Use Case** | **Completed** |
| Zip Client ID & Client Secret is configurable |  |
| Zip API Endpoints are configurable (To allow for Sandbox & Production Environment) |  |
| If applicable, Store ID is configurable |  |

Notes

### Transactions

There are a number of possibilities when transacting with Zip at checkout, from existing customers to new customers, as well as those applications that do not provide an instant result. These will need to match our spec for their respective scenarios to achieve certification. Each of these are listed below and must be handled for a complete integration.

|  |  |
| --- | --- |
| **Use Case** | **Completed** |
| POS provides a tender type of Zip |  |
| On selection of Zip, POS requests an “Instore Code” |  |
| POST a valid instore code to get a valid response |  |
| Exception: POST an invalid code to get an error msg |  |
| Exception: POST with a Duplicate Merchant Reference to get an error msg on POS |  |
| Exception: POST instore code to Zip and the request times out (10min) |  |
| POS begins polling and returns “Approved” state |  |
| Exception: POS begins polling and returns “Declined” state |  |
| On order completion by customer, POS should convey this to the operator |  |
| On order completion by customer, this can be verified via Zip Dashboard |  |
| Exception: Customer’s spend limit was exceeded for purchase |  |
| Exception: Network/Power failure on POS during transaction (Order Recovery) |  |
| **Refund** | |
| Operator can select a historical order and choose to refund it |  |
| Exception: Operator attempts to refund more than the order value |  |
| Operator can select the amount they wish to refund (partial/full refund) |  |
| Operator can process multiple partial refunds on an order |  |
| Cross-refunds are supported across stores (if applicable) |  |
| Online orders can be refunded through POS (if applicable) |  |
| **Transaction Data** | |
| Transaction data appears correctly on Zip Dashboard - API call contains as much data as possible (item name, amount, quantities, etc.) |  |
| Ensure the right item type (description, name, sku, price) is used for item details |  |
| Discount items - If gift card, voucher, store credit or coupons are accepted at checkout please make sure you can use it in conjunction with Zip |  |
| **Other** | |
| Exception: POS is configured with invalid credentials |  |
| Zip Order Number/Merchant Reference should be printed in the customers receipt |  |
| A guide has been prepared for operators on how to use Zip on your POS |  |
| Provide a video/album of a Zip purchase taking place (For channel partners) |  |
| Log all API calls (request, response, etc) |  |
| (Optional) Idempotency-Key is present in the header for transactional API requests (/pos/order, /pos/refund) |  |

Notes