API Workflow: Reward Store Management

Table of Contents

Summary

A great way to drive participation in a loyalty program is to offer a configurable Reward Store. Using the RewardStoresV2 and AcquisitionV2 APIs, both of which are part of the Offers domain, you can build and configure stores with a wide variety of controls. For instance, you can create stores that contain offers targeted to specific audiences or stores that contain deals for hidden menu items only available to certain members, thus enforcing exclusivity and motivating additional behavior. Stores can be set to run for a limited amount of time, or to be “evergreen” and never expire.

This workflow focuses on a typical sequence of events relating to reward stores that already exist; namely, allowing a customer to access offers from a reward store and then make a purchase via a specific offer.

Use Case Sequence

The sequence for this use case is:

1. Get reward store(s).
2. Get offers for reward store.
3. Purchase an offer.
4. If necessary, return an offer.

Data Configuration Prerequisites

This guide presumes that all offers are previously configured within the system. Furthermore, it presumes that reward stores are previously configured with relevant offers previously assigned to the store. These are one-time setup activities that are completed most efficiently in the SMP platform itself.

Step by Step with Examples

Featured in this workflow are the RewardStoresV2 and AcquisitionV2 APIs.

When issuing curl commands for these transactions, adhere to the following syntax:

- Begin each curl command with either `POST`.
- Specify: `-H 'Content-Type: application/json' -H 'authorization: Basic AUTH_ID'`
- Begin URL with same endpoint: `https://[ENDPOINT]/api/2.0/offers`

Step 1: Get Reward Store

The first step in making reward store offers available to customers for purchase is to retrieve the relevant reward store. Two endpoints for retrieving a reward store are available, depending upon how many stores exist.

Fetching a Single Reward Store

This endpoint allows you to retrieve the reward store definition, including reward store name, start date, end date, associated point accounts, associated eligibilities, and any custom flex field data:

```
POST /api/2.0/offers/rewardstores/get_reward_stores
```

The endpoint passes in a request object that identifies a single reward store.

Request

```
{
  "retailer_id": "string",
  "search_name": "string",
  "search_eligibility": [
    "string"
  ],
  "search_active_date": "string",
  "filters": [
    "string"
  ],
  "search_reward_store_ids": [
    "string"
  ],
  "sort_column": "string",
  "sort_direction": "string",
  "skip": 0,
  "take": 0
}
```

The platform returns a response object:

Response

```
{
  "response_payload": {
    "reward_stores": [
      {
        "id": "string",
        "internal_name": "string",
        "point_account_ids": [
          "string"
        ],
        "start_date": "2018-12-01T20:51:13.963Z",
        "end_date": "2018-12-01T20:51:13.963Z",
        "custom_data": "string",
        "eligibility_modes": [
          "string"
        ]
      }
    ],
    "total_rows": 0,
    "total_rows_unfiltered": 0,
    "skip": 0,
    "take": 0
  },
  "status": 0,
  "error_response": {
    "code": "string",
    "message": "string",
    "raw_message": "string",
    "stack_trace": "string"
  },
  "logs": [
    {
      "log_level": "Trace",
      "message": "string",
      "stack_trace": "string",
      "time_occurred": "2018-12-01T20:51:13.967Z",
      "duration": 0,
      "scope_observer_id": "00000000-0000-0000-0000-000000000000",
      "entity_id": "00000000-0000-0000-0000-000000000000"
    }
  ],
  "message": "string",
  "scope": {
    "retailer_id": "string",
    "event": "string",
    "extended_scope": {}
  },
  "request_payload": {},
  "authorized_integrator_id": "00000000-0000-0000-0000-000000000000",
  "started": "2018-12-01T20:51:13.967Z",
  "completed": "2018-12-01T20:51:13.967Z"
}
```

Note that the reward store ID in this response can be used in the endpoint detailed in Step 2: Get Reward Store Offers.

Fetching Eligible Reward Stores for a Customer

It’s not uncommon for an organization to maintain multiple reward stores. Perhaps they maintain one store for the general public and one for their platinum tier members. When this is the case, you can use this endpoint to discover the reward stores available to the customer:

```
POST /api/2.0/offers/rewardstores/get_user_reward_stores
```

The endpoint returns a list of eligible reward stores for a given customer. An organization may want to use reward store eligibility to develop multiple stores with different kinds of customers targeted. For example, an “invite-only” store or a reward store that is accessible to only members of a certain tier. In this case, you must first use this endpoint to determine the list of reward stores that a give user has access to.

The endpoint passes in the a request object that returns all eligible reward stores.

Request

```
{
  "retailer_id": "string",
  "user_id": "string",
  "sort_column": "string",
  "sort_direction": "string",
  "skip": 0,
  "take": 0,
  "include_point_balances": true
}
```

The platform returns a response object:

Response

```
{
  "response_payload": {
    "point_balances": [
      {
        "account_name": "string",
        "user_point_account_id": "string",
        "point_account_id": "string",
        "grouping_label": "string",
        "available_balance": 0,
        "life_time_value": 0
      }
    ],
    "reward_stores": [
      {
        "id": "string",
        "internal_name": "string",
        "point_account_ids": [
          "string"
        ],
        "start_date": "2018-12-11T22:25:41.001Z",
        "end_date": "2018-12-11T22:25:41.001Z",
        "custom_data": "string",
        "eligibility_modes": [
          "string"
        ]
      }
    ],
    "total_rows": 0,
    "skip": 0,
    "take": 0
  },
  "status": 0,
  "error_response": {
    "code": "string",
    "message": "string",
    "raw_message": "string",
    "stack_trace": "string"
  },
  "logs": [
    {
      "log_level": "Trace",
      "message": "string",
      "stack_trace": "string",
      "time_occurred": "2018-12-11T22:25:41.001Z",
      "duration": 0,
      "scope_observer_id": "00000000-0000-0000-0000-000000000000",
      "entity_id": "00000000-0000-0000-0000-000000000000"
    }
  ],
  "message": "string",
  "scope": {
    "retailer_id": "string",
    "event": "string",
    "extended_scope": {}
  },
  "request_payload": {},
  "authorized_integrator_id": "00000000-0000-0000-0000-000000000000",
  "started": "2018-12-11T22:25:41.001Z",
  "completed": "2018-12-11T22:25:41.001Z"
}
```

This response shows a payload that includes an ID in the *reward_stores* array for the reward store available to the customer. Note that this ID can be used in the endpoint detailed in Step 2: Get Reward Store Offers.

Step 2: Get Reward Store Offers

Next, get the offers associated with the eligible reward stores with the following endpoint:

```
POST /api/2.0/offers/rewardstores/get_reward_store_offers
```

The endpoint passes in the following request object that retrieves the offers within one or more reward stores.

Request

```
{
  "retailer_id": "string",
  "reward_store_ids": [
    "string"
  ],
  "limit_to_only_active": true,
  "exclude_free_offers": true,
  "include_expired": true,
  "skip": 0,
  "take": 0,
  "sort_column": "string",
  "sort_direction": "string",
  "culture": "string"
}
```

The request object can contain filters that can control for the following:

- Excluding inactive offers
- Excluding free offers
- Excluding expired offers

Furthermore, the *culture* attribute can determine which creative copy and image is to be returned with the request. This function allows clients to display reward store content in different languages.

After the endpoint makes the request for offers with reward store IDs, the platform returns a response object, which is shown below:

Response

```
{
  "response_payload": {
    "reward_store_offers": [
      {
        "id": "string",
        "root_offer_id": "string",
        "reward_store_id": "string",
        "price": 0,
        "start_date": "2018-12-01T20:51:13.997Z",
        "end_date": "2018-12-01T20:51:13.997Z",
        "offer_details": {
          "internal_name": "string",
          "title": "string",
          "description": "string",
          "terms": "string",
          "media": [
            {
              "id": "string",
              "offer_id": "string",
              "uri": "string",
              "category_id": "string",
              "category_name": "string",
              "content_type": "Image",
              "culture": "string"
            }
          ],
          "reward_store_media": {
            "id": "string",
            "offer_id": "string",
            "uri": "string",
            "category_id": "string",
            "category_name": "string",
            "content_type": "Image",
            "culture": "string"
          },
          "points": 0,
          "acquisition_start_date": "2018-12-01T20:51:13.997Z",
          "acquisition_end_date": "2018-12-01T20:51:13.997Z"
        }
      }
    ],
    "total_rows": 0,
    "skip": 0,
    "take": 0
  },
  "status": 0,
  "error_response": {
    "code": "string",
    "message": "string",
    "raw_message": "string",
    "stack_trace": "string"
  },
  "logs": [
    {
      "log_level": "Trace",
      "message": "string",
      "stack_trace": "string",
      "time_occurred": "2018-12-01T20:51:13.998Z",
      "duration": 0,
      "scope_observer_id": "00000000-0000-0000-0000-000000000000",
      "entity_id": "00000000-0000-0000-0000-000000000000"
    }
  ],
  "message": "string",
  "scope": {
    "retailer_id": "string",
    "event": "string",
    "extended_scope": {}
  },
  "request_payload": {},
  "authorized_integrator_id": "00000000-0000-0000-0000-000000000000",
  "started": "2018-12-01T20:51:13.998Z",
  "completed": "2018-12-01T20:51:13.998Z"
}
```

Now, with the offers displayed in a reward store, the offer can be issued and purchased by the customer.

Step 3: Purchase or Issue/Spend Offer

Offers can be purchased by customers. The purchase transaction occurs in two steps, the offer issued to the customer and the customer's point balance reduced by the cost of the offer. Programatically, this can be accomplished by:

- Making one call using the purchase endpoint to Issue offer and spend points in a single call. **This is the recommended approach**.
- Making two calls: One call that Issues an offer to a customer and then one that spends points via a subsequent purchase.

After successfully issuing the offer, it is available in the users offer wallet. The offer can be redeemed as part of a transaction or some other client-accepted redemption channel. If a client is using a third-party offer wallet, the offer purchase can be published to the third-party system to inform them of the offer issuance.

Purchasing an Offer

It is more efficient and recommended to leverage the purchase API. It may be efficient to have both the issuance and purchase of an offer achieved in one call. Before an offer is issued and a purchase is made, the system can verify that the customer has enough points in the account associated with the reward store ID to make a purchase. If they do, the offer is issued and the point balance is reduced by the appropriate amount.

This endpoint combines all of this functionality, issuing an offer to a validated point account and making a purchase that spends the customer's points:

```
POST /api/2.0/offers/rewardstores/purchase_offer
```

The endpoint passes in the purchase request object, which is shown below:

Request

```
{
  "retailer_id": "string",
  "reward_store_id": "string",
  "user_id": "string",
  "reward_store_offer_id": "string",
  "quantity": 0
}
```

The request object must contain a user ID, reward store offer ID, offer ID, and purchase quantity.

The platform returns a response object, which is shown below:

Response

```
{
  "response_payload": {
    "user_id": "string",
    "user_offer": {
      "offer_id": "string",
      "offer_group_id": "string",
      "offer_type": "string",
      "offer_custom_data": "string",
      "user_offer_custom_data": "string",
      "redemption_start_date": "2018-12-01T20:51:14.646Z",
      "redemption_end_date": "2018-12-01T20:51:14.646Z",
      "acquire_date": "2018-12-01T20:51:14.646Z",
      "redeem_date": "2018-12-01T20:51:14.646Z",
      "issued_by_offer_id": "string",
      "issued_by_user_offer_id": "string",
      "is_redeemable": true,
      "check_level": true,
      "pending_extended_data": true,
      "total_uses": 0,
      "remaining_uses": 0,
      "reward_xid": "string",
      "start_date": "2018-12-01T20:51:14.646Z",
      "end_date": "2018-12-01T20:51:14.646Z",
      "expiration_date": "2018-12-01T20:51:14.646Z",
      "id": "string",
      "root_offer_id": "string",
      "pos_discount_id": "string",
      "name": "string",
      "description": "string",
      "terms": "string",
      "culture": "string",
      "media": [
        {
          "id": "string",
          "offer_id": "string",
          "uri": "string",
          "category_id": "string",
          "category_name": "string",
          "content_type": "Image",
          "culture": "string"
        }
      ]
    },
    "child_user_offers": [
      {
        "offer_id": "string",
        "offer_group_id": "string",
        "offer_type": "string",
        "offer_custom_data": "string",
        "user_offer_custom_data": "string",
        "redemption_start_date": "2018-12-01T20:51:14.646Z",
        "redemption_end_date": "2018-12-01T20:51:14.646Z",
        "acquire_date": "2018-12-01T20:51:14.646Z",
        "redeem_date": "2018-12-01T20:51:14.646Z",
        "issued_by_offer_id": "string",
        "issued_by_user_offer_id": "string",
        "is_redeemable": true,
        "check_level": true,
        "pending_extended_data": true,
        "total_uses": 0,
        "remaining_uses": 0,
        "reward_xid": "string",
        "start_date": "2018-12-01T20:51:14.646Z",
        "end_date": "2018-12-01T20:51:14.646Z",
        "expiration_date": "2018-12-01T20:51:14.646Z",
        "id": "string",
        "root_offer_id": "string",
        "pos_discount_id": "string",
        "name": "string",
        "description": "string",
        "terms": "string",
        "culture": "string",
        "media": [
          {
            "id": "string",
            "offer_id": "string",
            "uri": "string",
            "category_id": "string",
            "category_name": "string",
            "content_type": "Image",
            "culture": "string"
          }
        ]
      }
    ],
    "points_spent": 0,
    "points_remaining": 0
  },
  "status": 0,
  "error_response": {
    "code": "string",
    "message": "string",
    "raw_message": "string",
    "stack_trace": "string"
  },
  "logs": [
    {
      "log_level": "Trace",
      "message": "string",
      "stack_trace": "string",
      "time_occurred": "2018-12-01T20:51:14.646Z",
      "duration": 0,
      "scope_observer_id": "00000000-0000-0000-0000-000000000000",
      "entity_id": "00000000-0000-0000-0000-000000000000"
    }
  ],
  "message": "string",
  "scope": {
    "retailer_id": "string",
    "event": "string",
    "extended_scope": {}
  },
  "request_payload": {},
  "authorized_integrator_id": "00000000-0000-0000-0000-000000000000",
  "started": "2018-12-01T20:51:14.646Z",
  "completed": "2018-12-01T20:51:14.646Z"
}
```

Issuing and Spending an Offer

If you want to issue an offer to a customer's offer wallet and then have that customer spend the point necessary to purchase it subsequently, use two calls. First, call this endpoint to issue the offer:

```
POST /api/2.0/offers/acquisition/issue
```

The endpoint passes in the user ID and offer ID in the request object, which is shown below:

Request

```
{
  "retailer_id": "string",
  "user_id": "string",
  "offer_id": "string",
  "quantity": 0,
  "issue_exact_version": true,
  "rollback_option": "AfterXFailedPasses"
}
```

The *quantity* attribute allows you to issue a multiplicity of offers. While the *rollback_option* specifies a number of issuance attempts before rolling back the request. Note that there is no point deduction in this request. It is simply a mechanism to issue an offer to a customer.

The platform returns a response object, which is shown below:

Response

```
{
  "response_payload": {
    "offer_id": "string",
    "root_offer_id": "string",
    "pos_discount_id": "string",
    "offer_custom_data": "string",
    "status_summary": "string",
    "issued_by_user_id": "string",
    "status_list": [
      {
        "user_id": "00000000-0000-0000-0000-000000000000",
        "user_offer_id": "00000000-0000-0000-0000-000000000000",
        "offer_id": "00000000-0000-0000-0000-000000000000",
        "root_offer_id": "00000000-0000-0000-0000-000000000000",
        "parent_user_offer_id": "00000000-0000-0000-0000-000000000000",
        "pending_extended_data": true,
        "fail_reason": "None"
      }
    ]
  },
  "status": 0,
  "error_response": {
    "code": "string",
    "message": "string",
    "raw_message": "string",
    "stack_trace": "string"
  },
  "logs": [
    {
      "log_level": "Trace",
      "message": "string",
      "stack_trace": "string",
      "time_occurred": "2018-12-01T20:51:10.936Z",
      "duration": 0,
      "scope_observer_id": "00000000-0000-0000-0000-000000000000",
      "entity_id": "00000000-0000-0000-0000-000000000000"
    }
  ],
  "message": "string",
  "scope": {
    "retailer_id": "string",
    "event": "string",
    "extended_scope": {}
  },
  "request_payload": {},
  "authorized_integrator_id": "00000000-0000-0000-0000-000000000000",
  "started": "2018-12-01T20:51:10.937Z",
  "completed": "2018-12-01T20:51:10.937Z"
}
```

After the offer has been issued to the customer, a new request can spend the points in the customer’s account. For this step, use the “spend” endpoint, which is in the Incentives domain’s UserPoints API. This endpoint is detailed in the Decreasing a Point Balance section of *API Workflow: User Points Management*.

Step 4: Return an Offer

Customers can choose to return an offer. This endpoint revokes the offer from a customer’s wallet and restores the points to their account:

```
POST /api/2.0/offers/acquisition/return
```

The endpoint passes in a request object that must include the user offer ID.

Request

```
{
  "retailer_id": "string",
  "user_offer_id": "string",
  "returned_by": "string",
  "reason": "string"
}
```

Note that the request can also contain who is requesting the return along with the reason for the return. The platform returns a response object, which is shown below:

Response

```
{
  "response_payload": {
    "user_id": "string",
    "user_offer_id": "string",
    "offer_id": "string",
    "root_offer_id": "string",
    "pos_discount_id": "string",
    "returned_by": "string",
    "reason": "string",
    "points_returned": 0
  },
  "status": 0,
  "error_response": {
    "code": "string",
    "message": "string",
    "raw_message": "string",
    "stack_trace": "string"
  },
  "logs": [
    {
      "log_level": "Trace",
      "message": "string",
      "stack_trace": "string",
      "time_occurred": "2018-12-01T20:51:11.137Z",
      "duration": 0,
      "scope_observer_id": "00000000-0000-0000-0000-000000000000",
      "entity_id": "00000000-0000-0000-0000-000000000000"
    }
  ],
  "message": "string",
  "scope": {
    "retailer_id": "string",
    "event": "string",
    "extended_scope": {}
  },
  "request_payload": {},
  "authorized_integrator_id": "00000000-0000-0000-0000-000000000000",
  "started": "2018-12-01T20:51:11.138Z",
  "completed": "2018-12-01T20:51:11.139Z"
}
```