



Seller Center (SC) API

October 2016

Agenda

New SC Seller-facing API

- ✓ **What's API?**
- ✓ Getting Access
- ✓ Existing vs. New SC API
- ✓ Process Flow for API - Product Listing & Order Processing
- ✓ Support Setup



What is API? What are the benefits?

API - Application Programming Interface.

Basically integration from seller's backend system to our system.

2 main functions - GET info, POST info.

Benefits

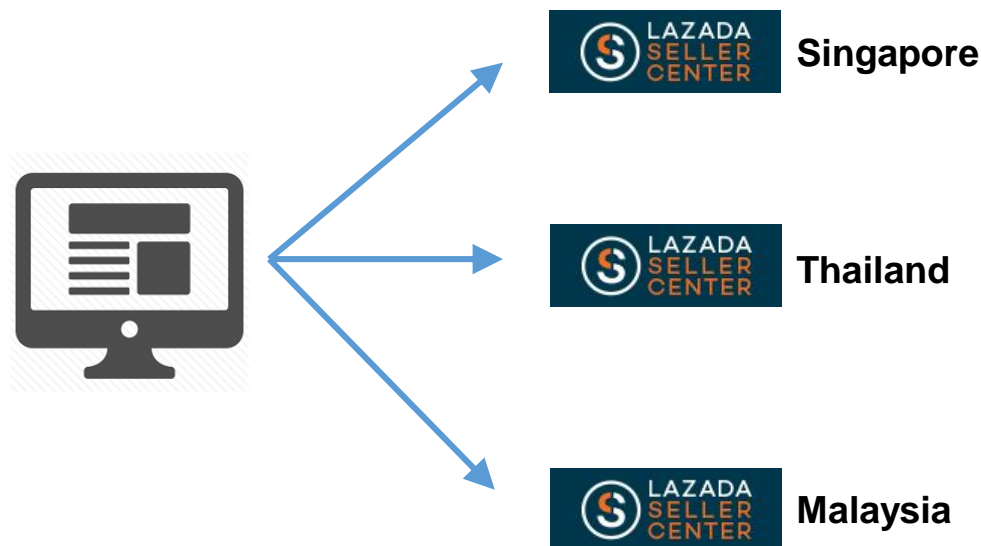
- 1 Sellers do not need to maintain >1 system for product publishing and order management
- 2 Sync for stock, orders, products, more

Getting Access to SC API

Getting Access

System and API Endpoint URLs

- Seller has to integrate his system with each Seller Center independently by using the same API specification.
- Each Seller Center has unique URL and respective set of API keys for users.



SG Sandbox for Test Phase

System: <http://seller.sgsbx.ali-lazada.com>

API Endpoint: <https://api.sgsbx.ali-lazada.com>

SG LIVE environment

System: <http://sellercenter.lazada.sg>

API Endpoint: <https://api.sellercenter.lazada.sg>

TH Sandbox for Test Phase

System: <http://asc-staging.sellercenter.lazada.co.th>

API Endpoint: asc-staging-api.sellercenter.lazada.co.th

TH LIVE environment

[Add]

MY Sandbox for Test Phase

System: asc-staging.sellercenter.lazada.com.my

API Endpoint: asc-staging-api.sellercenter.lazada.com.my

MY LIVE environment

[Add]

Getting Access

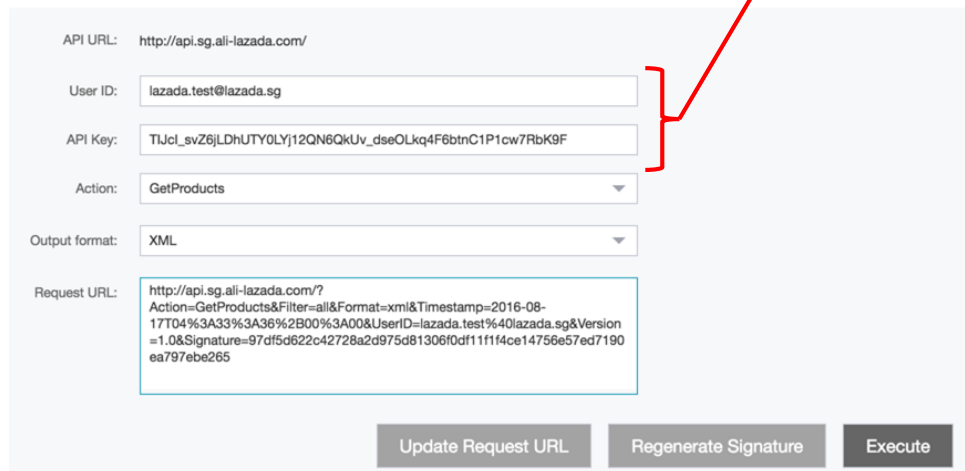
Retrieving API Credentials

1. Via User Management

- Go to **Manage User** & retrieve the corresponding API key for respective API user

2. Via API Explorer

- Go to **API Reference**
- Scroll down to **API Explorer** where your User ID and API key can be found



API URL: `http://api.sg.ali-lazada.com/`

User ID: `lazada.test@lazada.sg`

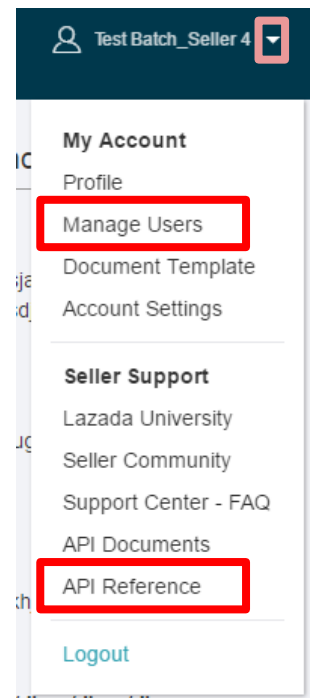
API Key: `TlJcI_svZ6jLDhUTY0LYj12QN6QkUv_dseOLkq4F6btrC1P1cw7RbK9F`

Action: `GetProducts`

Output format: `XML`

Request URL: `http://api.sg.ali-lazada.com/?Action=GetProducts&Filter=all&Format=xml&Timestamp=2016-08-17T04%3A33%3A36%2B00%3A00&UserID=lazada.test%40lazada.sg&Version=1.0&Signature=97df5d622c42728a2d975d81306f0df11f1f4ce14756e57ed7190ea797ebe265`

Update Request URL Regenerate Signature Execute



Getting Access

Using API Explorer

- API Explorer allows users to easily execute API calls to the Seller Center system via our web interface.
- Request URL and request body (XML) (where payload is necessary) will be populated for easy reference.
- User can change, add, delete parameters and attributes before executing the API call. The parameters (includes Action, Format, Timestamp, UserID, Version) should be arranged in alphabetical order in the Request URL and “Signature” is to be included at the end.
- User must regenerate signature before executing the API call if he/she changed any parameters in the URL.

API URL:

User ID:

API Key:

Action:

Output format:

Request URL:

Once you have chosen the action, the Request URL and XML (where applicable) appears immediately.

Getting Access

API Documentation

Documentation comprising all details about available endpoints, requests and responses format, error listing, approach for signing request & more. <https://lazada-sellercenter.readme.io>



Documentation Support

v1.0 Documentation

OVERVIEW

Introduction to the SellerCenter API

Requests and Responses

Errors

Roles and Privileges

Signing Requests

PRODUCT ENDPOINTS

GetProducts

CreateProduct

UpdateProduct

RemoveProduct

SetImages

UploadImage

MigrateImage

UpdatePriceQuantity

Introduction to the SellerCenter API

You'll be up and running in no time...

SellerCenter allows a store keeper to manage the products and orders in their online store.

Accordingly, the SellerCenter API enables the programmatic maintenance of products and orders.

E.g., you might use the API to import products, or to migrate orders into your accounting system.

Communication Format

Generally speaking, all requests originate with you; in other words, you contact the API. The API does not contact you.


Existing vs. New SC API

Existing vs. New SC API

Request and Response Format

In new Seller Center, both XML and JSON are supported.

- In the output format, JSON or XML will be returned ([URL](#)).
- If you are doing a call via POST with additional data in request body (payload), these data must be in XML format, regardless of the chosen output format (as shown in API Explorer).



The screenshot shows the API Explorer interface with the following fields:

- Output format:** A dropdown menu set to "JSON".
- Request URL:** A text box containing the URL: `http://api.sg.ali-lazada.com/?Action=CreateProduct&Format=json&Timestamp=2016-08-21T16%3A29%3A32%2B00%3A00&UserID=lazada.api3%40mailinat.or.com&Version=1.0&Signature=4c6e08d3038a6788042bdc2625ef1824d132272b60421c8d0a1424ac744bf99a`
- Request body (XML):** A text box with a red border containing the XML payload:

```
<Attributes>
  <name>api create product test sample</name>
  <short_description>This is a nice
product</short_description>
  <brand>Remark</brand>
  <model>asdf</model>
```

- For most importance, please read [this page](#) to do this by different coding languages.

Existing vs. New SC API

Feed Concept - No Longer Applicable

In the new Seller Center, the API calls are synchronous (instead of asynchronous in the existing SC).

What's the impact?

- Product creation, update and removal via API will be processed during that instance of API call ('immediately'). There is no feed queueing concept. This means that your application will also not execute other calls, before a response is returned by the API for the current call.
- Feed endpoints will no longer be available as it's not applicable in the new SC. There is no need to retrieve success status of your product creation/update/removal via get FeedStatus.
- 'Instantaneous' response will be given via API response.
- Note that all your actions will be immediate. There is no way to cancel the product creation or update via API.

Example of successful product creation via API

```
{
  "SuccessResponse": {
    "Head": {
      "RequestId": "",
      "RequestAction": "CreateProduct",
      "ResponseType": "Product",
      "Timestamp": "2016-08-21T04:43:57+0000"
    },
    "Body": {
      "Warnings": []
    }
  }
}
```

Existing vs. New SC API

Available API Endpoints

Existing SC API	New SC API
Product Endpoints	
GetProducts - - - ProductCreate ProductUpdate Image - ProductRemove GetBrands GetCategoryTree GetCategoryAttributes GetCategoriesByAttribute	GetProducts (Updated) SearchSPUs (NEW) UploadImage (NEW) MigrateImage (NEW) CreateProduct (Updated) UpdateProduct (Updated) SetImages (Updated) UpdatePriceQuantity (NEW) RemoveProduct (Updated) GetBrands (No changes) GetCategoryTree (No changes) GetCategoryAttributes (No changes) <i>Removed (No longer applicable)</i>
Quality Control Endpoints	
GetQcStatus (No changes)	GetQcStatus (No changes)

Existing vs. New SC API

Available API Endpoints

Existing SC API	New SC API
Sales Order Endpoints	
GetOrders	GetOrders (No changes)
GetOrder	GetOrder (No changes)
GetOrderComments	<i>Removed (No longer applicable)</i>
GetOrderItems	GetOrderItems (No changes)
GetMultipleOrderItems	GetMultipleOrderItems (No changes)
SetStatusToCanceled	SetStatusToCanceled (No changes)
SetStatusToPackedByMarketplace	SetStatusToPackedByMarketplace (No changes)
SetStatusToReadyToShip	SetStatusToReadyToShip (No changes)
SetStatusToShipped	<i>Removed (N.A. for seller)</i>
SetStatusToFailedDelivery	<i>Removed (N.A. for seller)</i>
SetStatusToDelivered	<i>Removed (N.A. for seller)</i>
SetInvoiceAccessKey	<i>Removed (Not applicable)</i>
GetDocument	GetDocument (No changes)
GetFailureReasons	GetFailureReasons (No changes)
SetInvoiceNumber	SetInvoiceNumber (No changes)

Existing vs. New SC API

Available API Endpoints

Existing SC API	New SC API
Shipment Provider Endpoints	
GetShipmentProviders	GetShipmentProviders (No changes)
Seller Endpoints	
GetMetrics	GetMetrics (No changes)
GetPayoutStatus	GetPayoutStatus (No changes)
GetStatistics	GetStatistics (No changes)
SellerUpdate	SellerUpdate (No changes)
UserUpdate	UserUpdate (No changes)

Existing vs. New SC API

Available API Endpoints

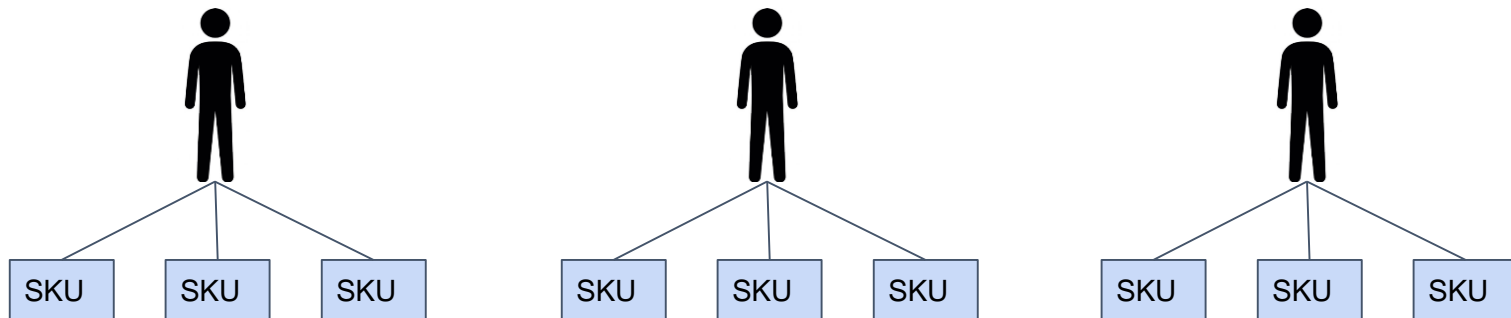
Existing SC API	New SC API
Feed Endpoints	
FeedList	<i>Removed (No longer applicable)</i>
FeedOffsetList	<i>Removed (No longer applicable)</i>
FeedCount	<i>Removed (No longer applicable)</i>
FeedCancel	<i>Removed (No longer applicable)</i>
GetFeedRawInput	<i>Removed (No longer applicable)</i>
FeedStatus	<i>Removed (No longer applicable)</i>
Manifest Endpoints	
GetManifestList	<i>Removed (N.A. for Lazada)</i>
CreateForwardManifest	<i>Removed (N.A. for Lazada)</i>
GetManifestDocument	<i>Removed (N.A. for Lazada)</i>
SetManifestStatusToShipped	<i>Removed (N.A. for Lazada)</i>

Product Listing

Current Concept - Seller SKU

Today SC product management is based on Seller SKUs (stock keeping unit). SKUs are distinct sale items unique to each seller. In reality, SKU is an inventory unit, primarily used for stock management.

Product attributes, such as model and brand are manually filled by each seller.



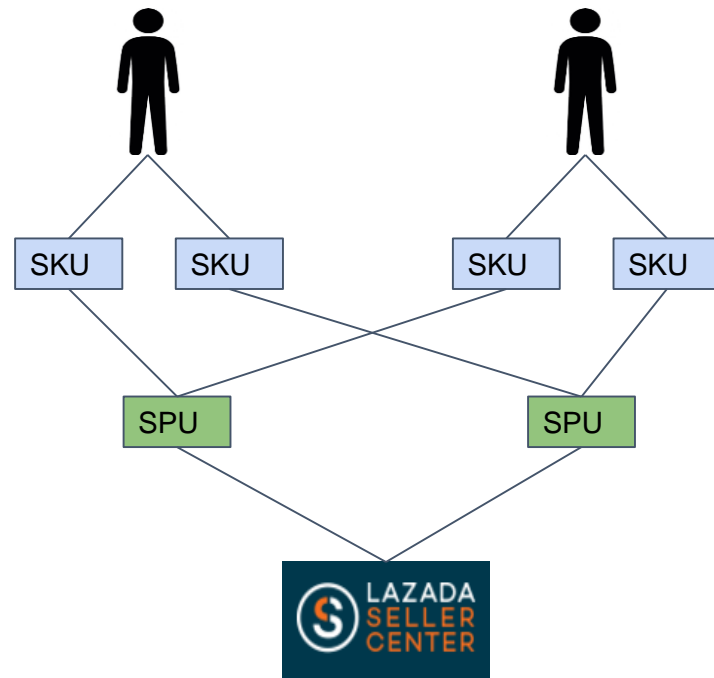
Product Listing

New Concept: SPU and SKU

SPU (Standard Product Unit) is a master product. This master product contains a collection of attributes (e.g. brand, model) that are common to a product across sellers.

Product attributes for each SPU is centrally managed by Lazada. Multiple seller SKUs can be associated to a SPU. SKUs contains more detailed attributes that are specific to each seller (e.g. price, image, package info).

What we currently call “*variations*” in SC will be treated as individual SKUs in the new SC.



Example

SPU - Apple iPhone 6

SKU - Apple iPhone 6 Silver 16GB, Silver 64GB, Black 16GB, Black 64GB

Benefits of SPU

Efficient Content Management

For common products, sellers will no longer need to fill in some product attributes (belonging to SPU). Thus, enabling efficient product creation process

Separating Stock and Product Management

Letting Lazada manage master product information means sellers can focus more on selling and inventory control

Improving Frontend Search Results

A centralized product database means that your product will be more easily searchable and benchmarked against similar products

Product API Endpoints

Attribute Requirements (Payload Structure)

Data requirements for products differ based on its corresponding category. E.g. a product under Home & Living category will have different attribute sets from a product under Fashion category.

To ensure completeness of your product data, use the following:

1. Via API

- Use [GetCategoryTree](#) method call to retrieve category IDs
- Use [GetCategoryAttributes](#) method call to retrieve attribute requirements for a specific category ID

2. Via Web Interface

Refer to [Product API Examples](#) for each category (under section “API Reference”)

```
▼ Computers & Laptops

Minimum API Call
<PrimaryCategory></PrimaryCategory>
<Attributes>
  <name></name>
  <short_description></short_description>
  <brand></brand>
</Attributes>
<Skus>
  <sku>
    <sellerSKU>ABC-1000-202</sellerSKU>
    <price></price>
    <package_weight></package_weight>
    <package_content></package_content>
    <package_height></package_height>
    <package_length></package_length>
    <package_width></package_width>
  </sku>
</Skus>

Full API Call
<PrimaryCategory></PrimaryCategory>
<SPUID></SPUID>
<Attributes>
  <name></name>
  <description></description>
  <short_description></short_description>
  <video></video>
</Attributes>
```

Tips

Field “Description” can contain certain HTML tags, including ul, li and span. If HTML is embedded, it must be escaped as character data (below in **green**). Note that table format is not accepted for this field.

- <Description><![CDATA[la descripción
negrita]]></Description>

Existing vs. New SC API

Data Limitation

More details to come

Process Flow

Process Flow

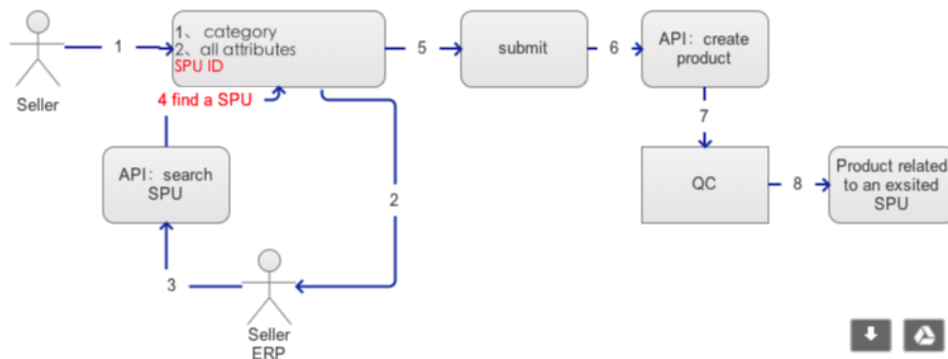
Product Listing with new SPU Concept

New product management flow allows user to search for an appropriate SPU for a SKU to be created

→ If appropriate SPU is returned, the SPU contain all necessary master product attributes. By using the SPU relation, seller doesn't have to fill certain master product attributes.

→ If SPU cannot be found, seller has to create product tagged to a certain subcategory. The seller has to fill in all attributes. During Content Quality Control review, Lazada team will tag this product to an existing SPU or create a new SPU accordingly.

More details in following slides.

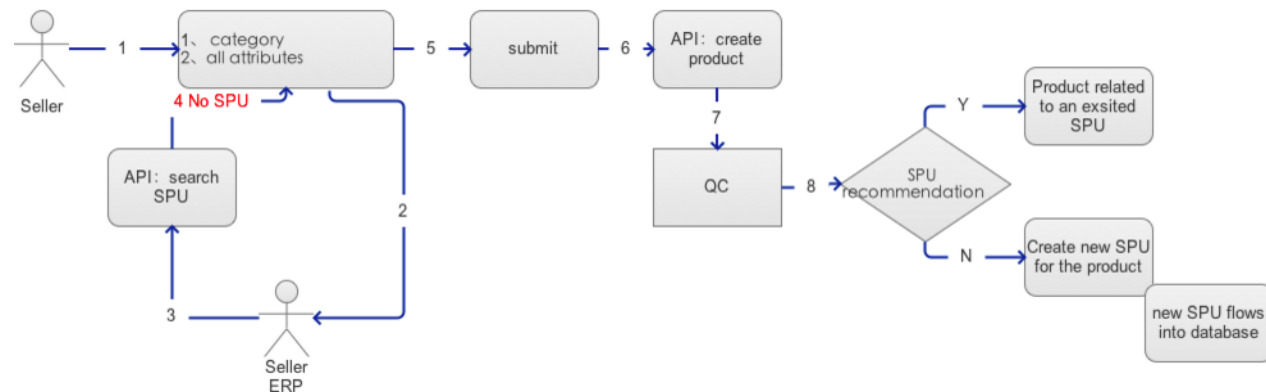


Flow 1: Seller finds relevant SPU

- Seller searches for SPU
- Seller creates SKU with appropriate SPU
- After SKU Quality Control is approved, SKU is created with the selected SPU

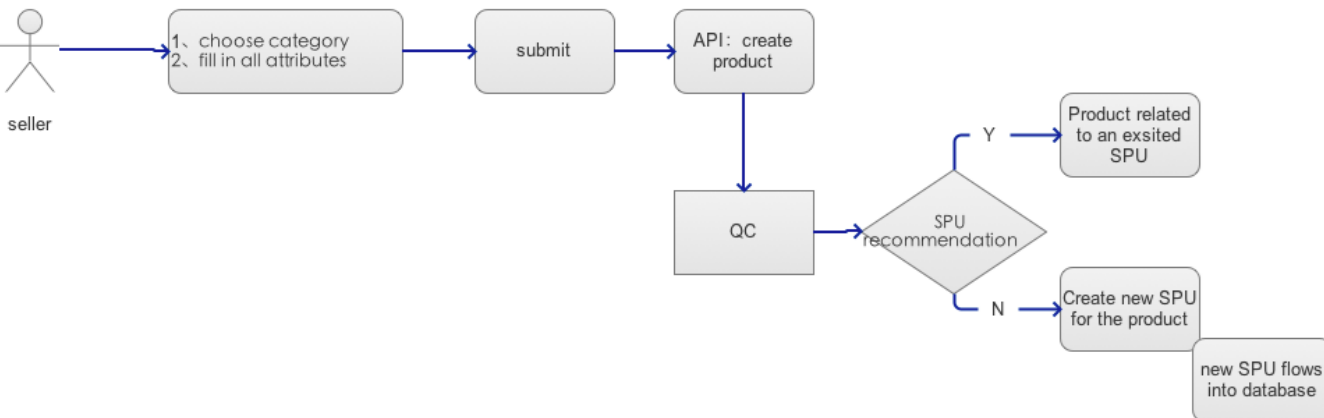
Process Flow

Product Listing with new SPU Concept



Flow 2: Seller can't find relevant SPU

- Seller searches for SPU
- Seller does not find appropriate SPU
- Seller search for suitable subcategory, downloads category attributes, and creates product
- Lazada Quality Control will recommend appropriate SPU for product, or create a new SPU if it does not yet exist



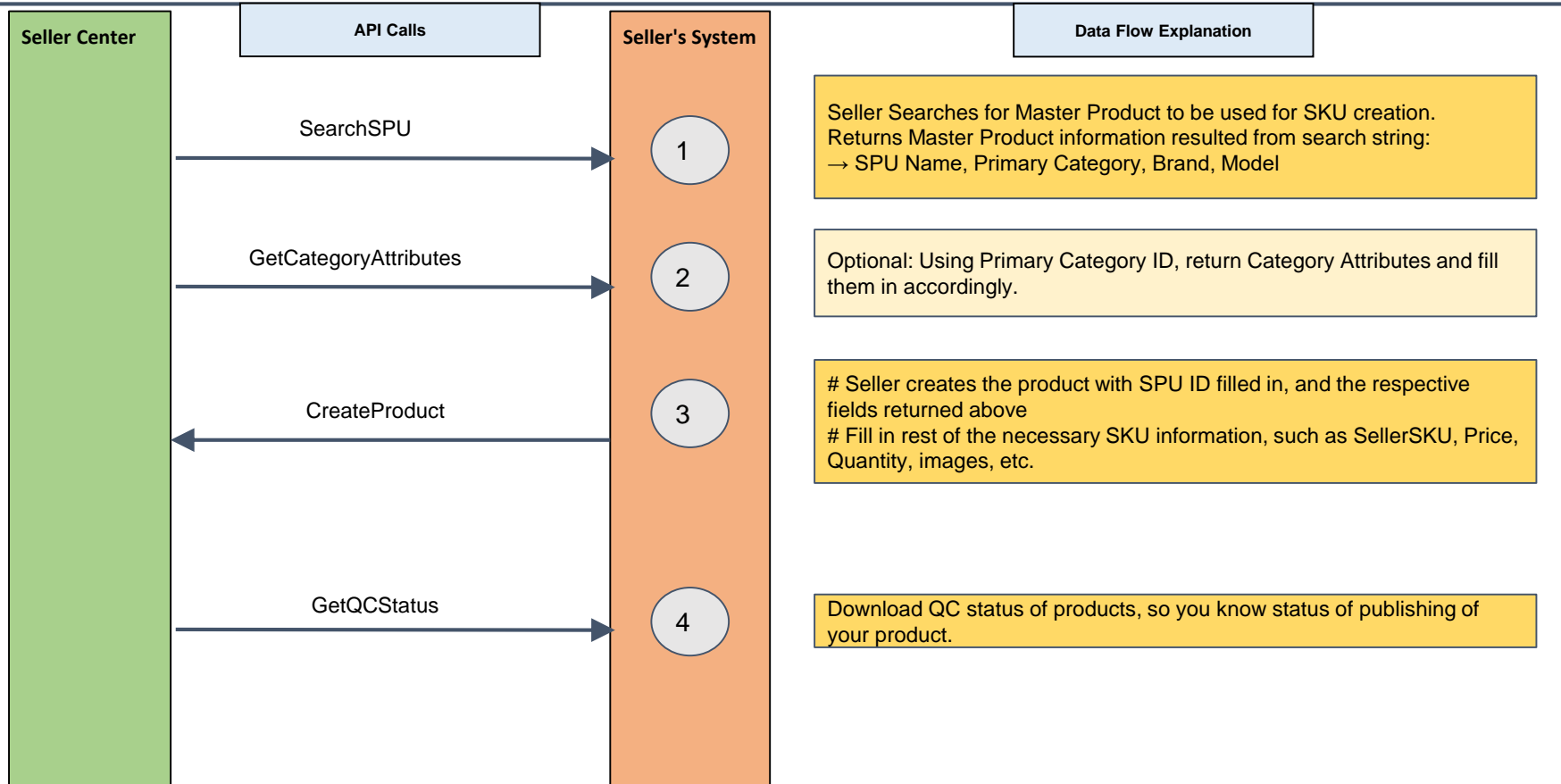
Flow 3: Seller creates SKU without SPU

- Seller search for suitable subcategory, downloads category attributes, and creates product directly
- Lazada Quality Control will recommend appropriate Master Product for product, or create a new Master Product if it does not yet exist

Process Flow

Product Listing 1: Seller finds relevant SPU

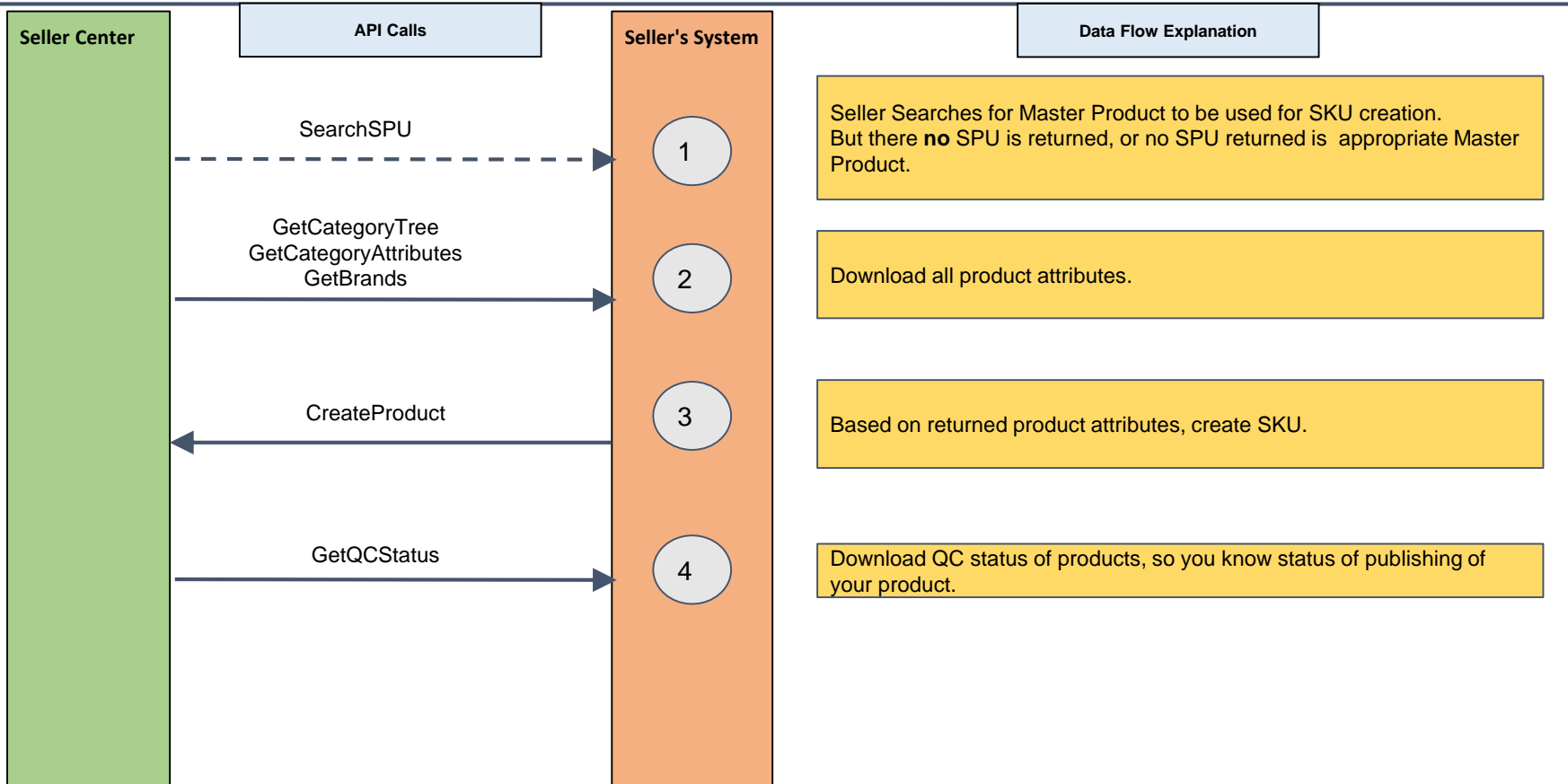
→ * Arrows indicate data flow



Process Flow

Product Listing 2: Seller can't find relevant SPU

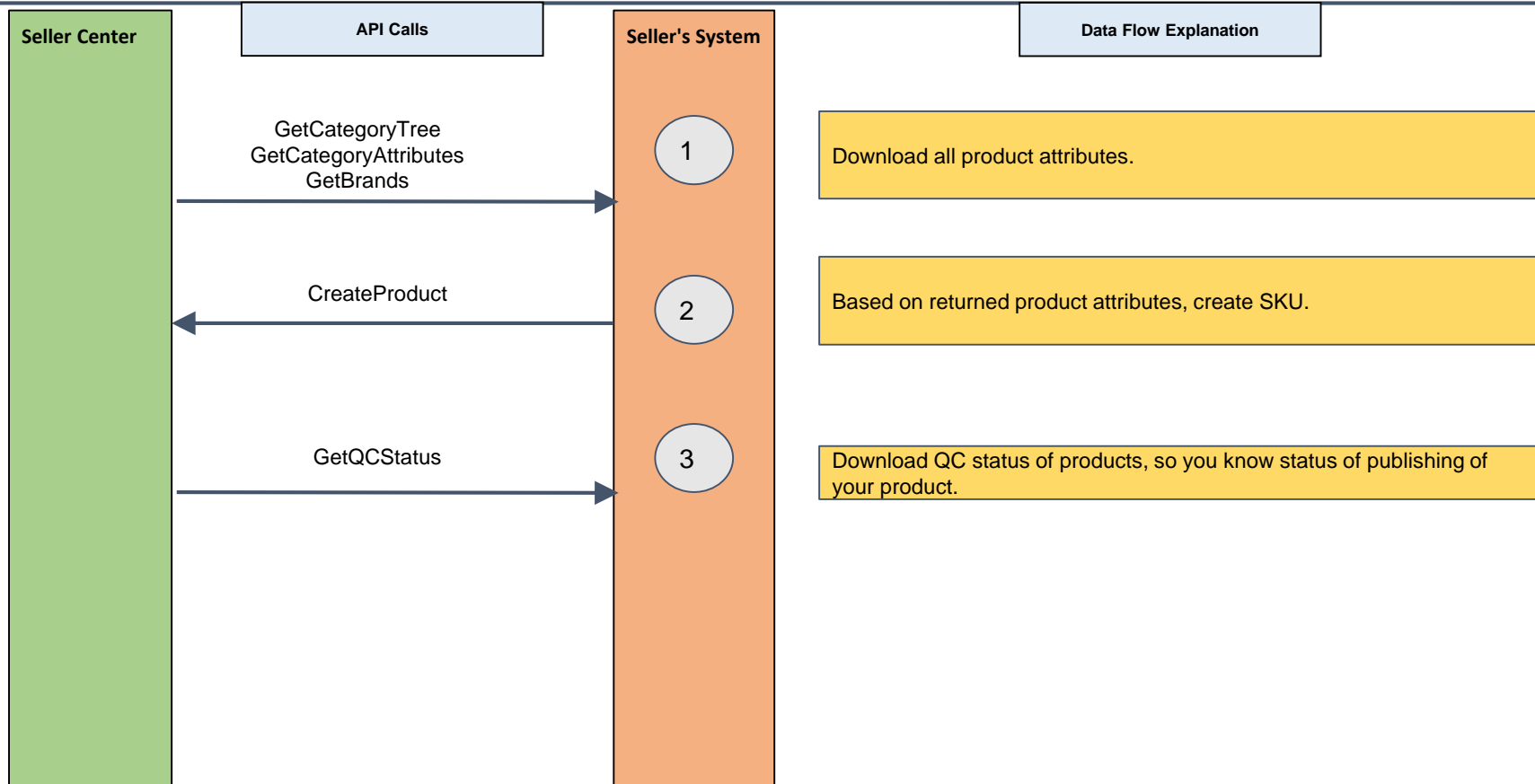
→ * Arrows indicate data flow



Process Flow

Product Listing 3: Seller creates product without SPU

→ * Arrows indicate data flow



Process Flow

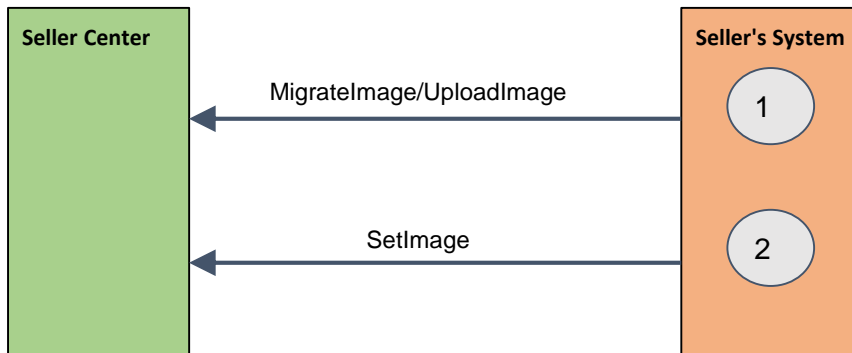
Product Listing

→ * Arrows indicate data flow

Image Management

API Calls

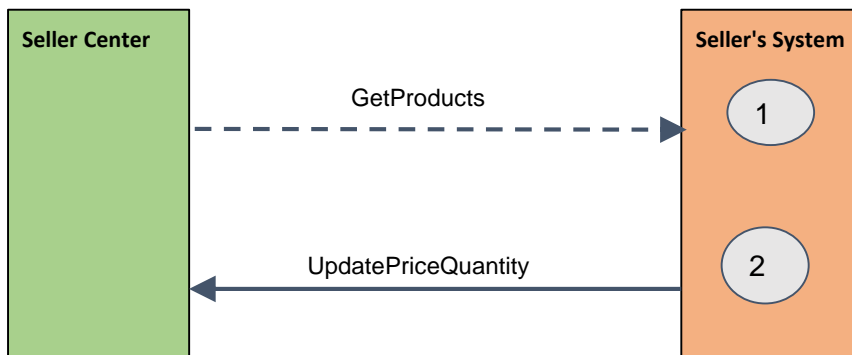
Data Flow Explanation



Migrate an online image from another website to our server, or
Upload image from your file on local drive
→ Image URL from our server will be returned

SetImage for the required SKU. Max 8 images.

Price/Quantity Update



Retrieve latest product information in SC account

Update price, sales price, sales start & end date, and quantity of
respective SKU

Order Processing - Choosing a suitable model

Order related API calls are backward compatible, meaning that the call structures and behavior remain the same. No change to **naming** of the API calls.

Based on seller's order processing model, please select the suitable integration for your company

Seller Operational Model	API integration model
<p>Seller perform order processing and confirm shipping by a single process / person</p> <ul style="list-style-type: none">• Download shipping label• Tracking Number• Apply label on parcel• Confirm Ready to Ship (RTS)	Model A: single step integration
<p>Seller has 2 different persons/processes involve order process and confirm shipping separately</p> <ul style="list-style-type: none">• Step 1<ul style="list-style-type: none">• Download shipping label• Tracking Number• Apply label on parcel• Step 2<ul style="list-style-type: none">• Confirm Ready to Ship (RTS)	Model B: 2 steps integration

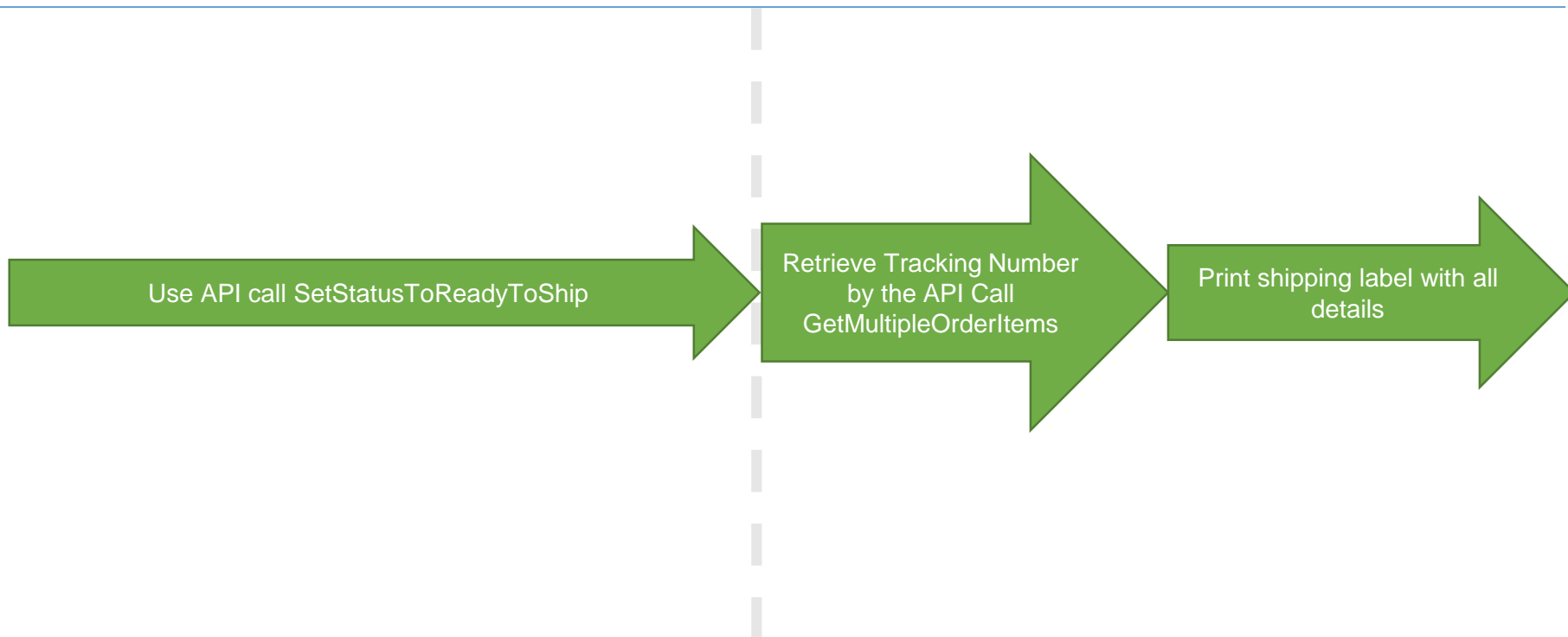
Process Flow

Order Processing - Choosing a suitable model

*Model A: **single** process/person to do order processing and confirm shipping*

Pending

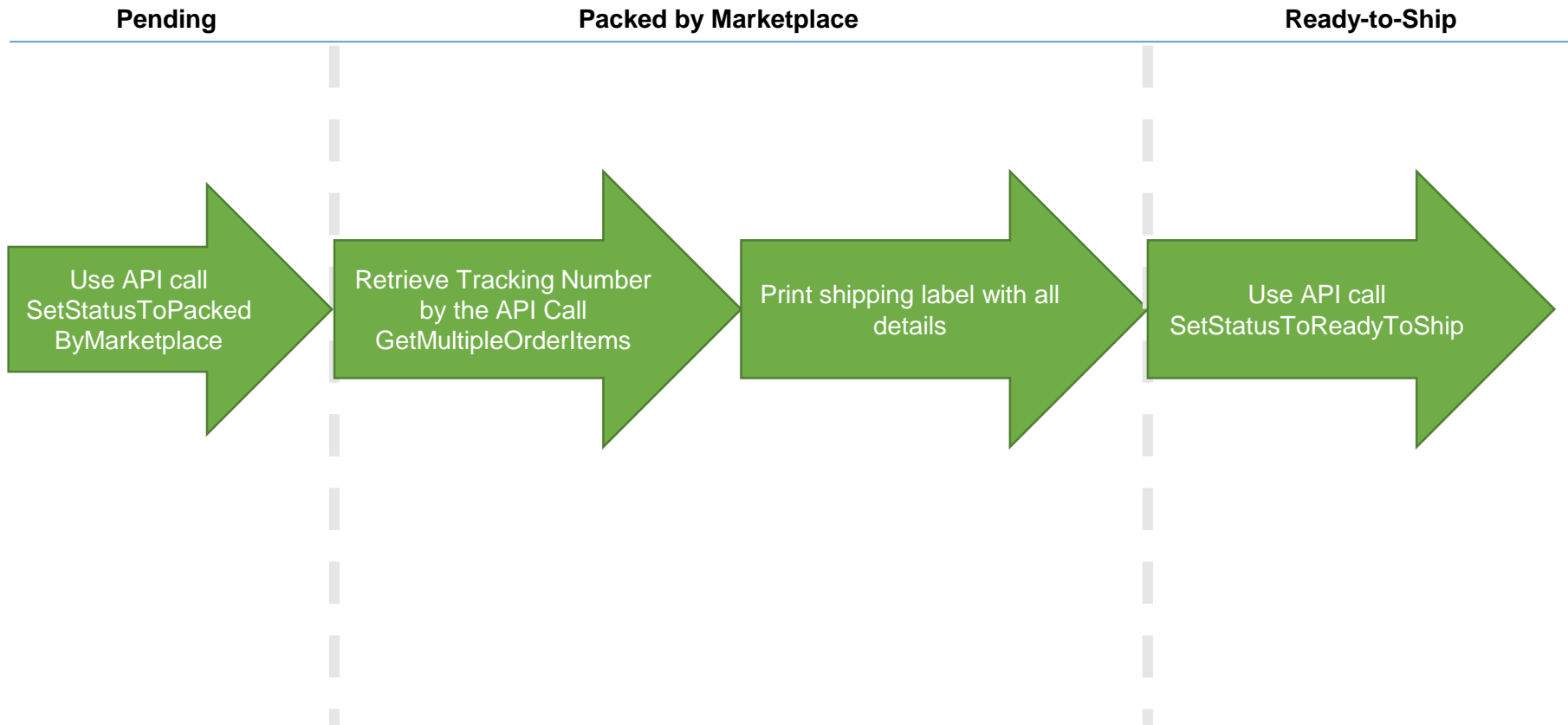
Ready-to-Ship



Process Flow

Order Processing - Choosing a suitable model

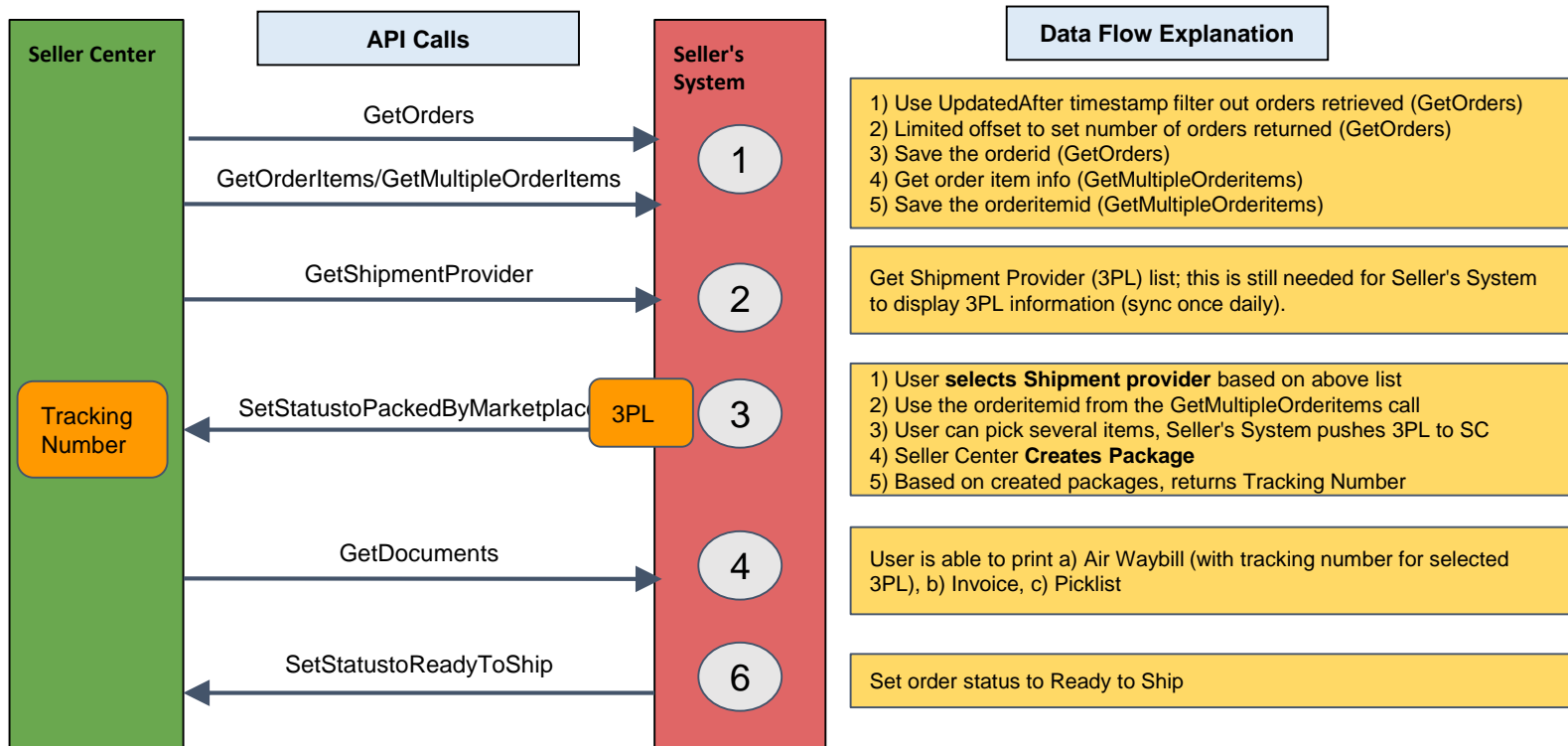
Model B: 2 processes/persons to do order processing and confirm shipping



Process Flow

Order Processing - Seller selects 3PL, auto-generated TN

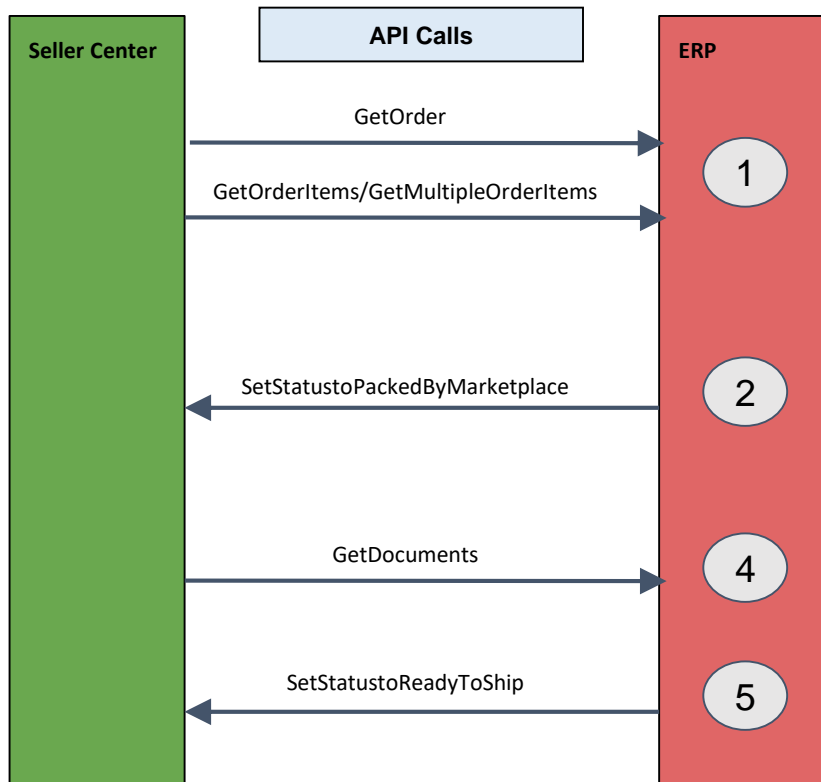
→ * Arrows indicate data flow



Process Flow

Order Processing - Touch-free Shipping

→ * Arrows indicate data flow



Data Flow Explanation

- 1) Use UpdatedAfter timestamp filter out orders retrieved (GetOrders)
- 2) Limited offset to set number of orders returned (GetOrders)
- 3) Save the orderid (GetOrders)
- 4) Get order item info (GetMultipleOrderitems)
- 5) Save the orderitemid (GetMultipleOrderitems)

Set a backend logic in ERP to set orders to packed by marketplace.
> ShippingProvider parameter = (empty)
Based on created packages, returns:

- 1) Tracking Number
- 2) Shipping Provider (Shipping Provider is preselected in Seller Center)

User is able to print a) Shipping Label , b) Invoice

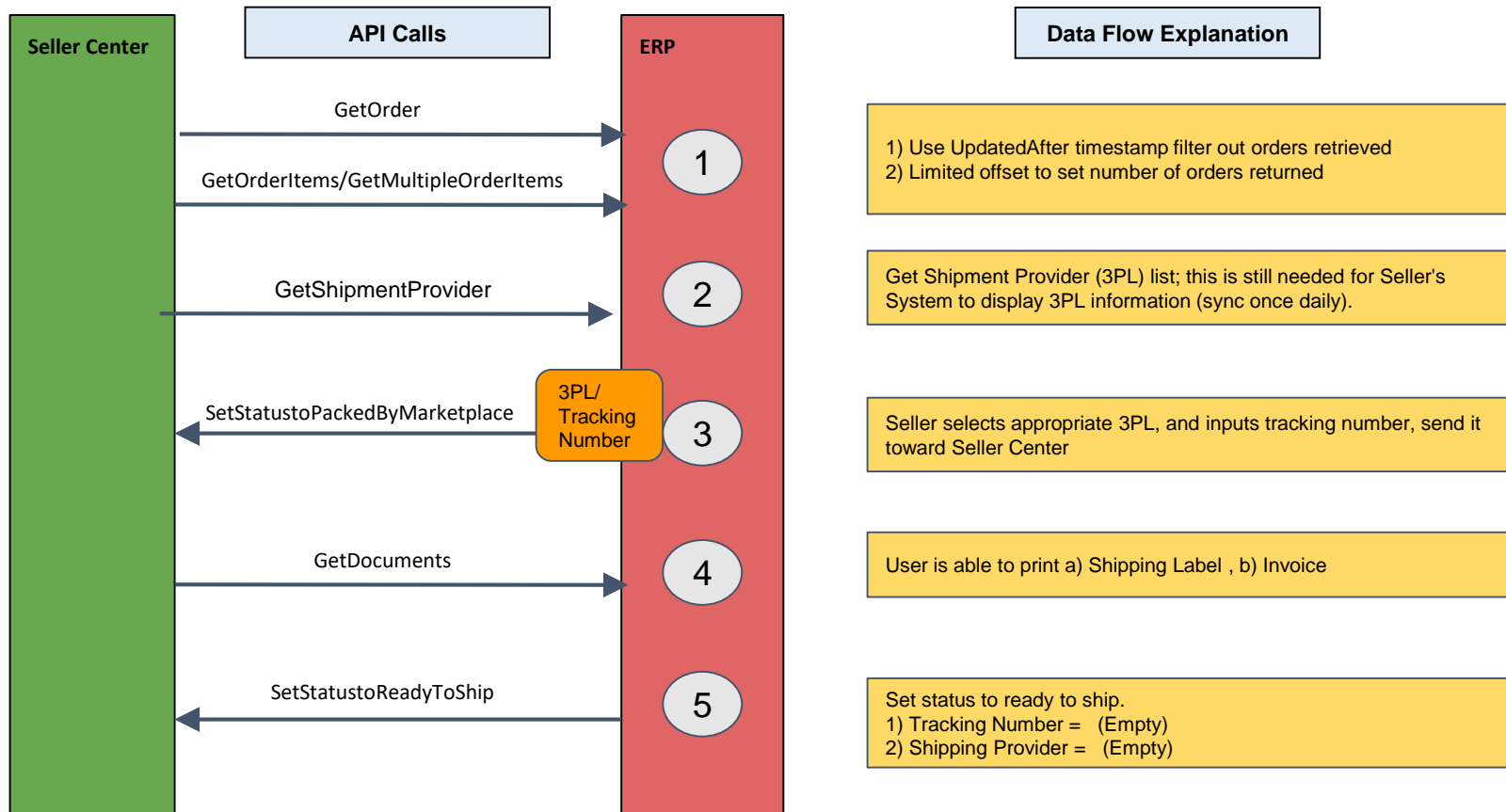
Set status to ready to ship.

- 1) Tracking Number = (Empty)
- 2) Shipping Provider = (Empty)

Process Flow

Order Processing - Seller sets Tracking Number

→ * Arrows indicate data flow



Development Kit

Development Kit

Sandbox Environment & SDKs

1. There's an available sandbox environment to facilitate your test phase during the integration project.
 - a. Access details to be released soon

1. We offer various Software Development Kit (SDK) for commonly used web programming languages in order to ease the adoption of our API. Feel free to use it and contribute to them since the code is published under Open Source MIT Licence.
 - a. Java SDK: In Progress

Support

Contact us at <https://lazada.formstack.com/forms/lazadascapi>

Summary

Summary

- ✓ API will be available in new SC platform
- ✓ Most endpoints are backward compatible
- ✓ Some changes for Product API (SPU and Image)
- ✓ Feed API and Manifest API will no longer be available
- ✓ API is synchronous rather than asynchronous



Thank you