# **Postman Collection Documentation for PriceLabs APIs**

## **1. Overview**

The **PriceLabs Postman Collection** is a structured suite of API requests designed to test key functionalities of the PriceLabs platform, including organization data retrieval, notification management, bedroom count updates, pricing adjustments, and custom pricing setups. These APIs ensure seamless communication and data flow across the application, allowing users to interact with core PriceLabs services efficiently.

### **Environment**

* **Environment Name:** pricelabs\_prod
* **Base URLs:**
  + API URL: {{url}}
  + Web URL: {{web\_url}}
  + DSO URL: {{dso\_url}}
* **Authentication:** Bearer token ({{token}}) for secure API access.

## **2. Important Note**

* This collection **does not cover all possible scenarios** for PriceLabs APIs.
* **Validation Issues:** The **DSO API** and **min\_price API** exhibit incomplete or improper validations for some fields. However, client-side validations are implemented for these scenarios.
* These limitations should be considered when using this collection to test or evaluate API functionality.

## **3. Collection Structure**

### **3.1. Requests**

The collection comprises several API requests, each serving a specific function. Below are the highlights:

#### **1. Get Organization**

* **Description:** Fetches organization details.
* **Method:** GET
* **Endpoint:** {{url}}/v1/widgets/organization
* **Tests Implemented:**
  + Verify response status code is 200.
  + Validate the organization name in the response body matches "PriceLabs".

#### **2. Notification Management**

* **Description:** Retrieves unseen notifications with a limit of 100.
* **Method:** GET
* **Endpoint:** {{url}}/v1/widgets/notifications/unseen?limit=100
* **Tests Implemented:**
  + Verify response status code is 200.
  + Ensure the notification count matches the expected value (1).

#### **3. Update Bedroom Count (Positive and Negative Scenarios)**

* **Description:** Updates the bedroom count for a given listing.
* **Method:** POST
* **Positive Case Endpoint:** {{web\_url}}/update\_bedroom\_count
  + **Validation:** Response status is 200, and the success message is verified.
* **Negative Case Endpoint:** {{web\_url}}/update\_bedroom\_count
  + **Validation:** Response status is 400, and the error message "Listing not found" is verified.

#### **4. Update Minimum Price**

* **Description:** Updates the minimum price for a specific listing.
* **Method:** POST
* **Endpoint:** {{web\_url}}/update\_price
* **Validation:** Confirms response status is 200.

#### **5. Add Custom Pricing**

* **Description:** Adds custom pricing details for a specific property listing.
* **Method:** POST
* **Endpoint:** {{dso\_url}}/api/add\_custom\_pricing
* **Validation:** Response status is 200.

## **4. Environment Configuration**

The collection utilizes the following environment variables to ensure flexibility and maintainability:

| **Variable Name** | **Value** | **Description** |
| --- | --- | --- |
| token | Token for API authentication | Access control via JWT token. |
| url | https://api.novu.co | Base API URL. |
| web\_url | https://pricelabs.co | Web application base URL. |
| dso\_url | https://app.pricelabs.co | DSO-specific API base URL. |

## **5. Testing Methodology**

Each request contains pre-written **Postman Test Scripts** to validate API responses:

* **Status Code Validation:** Ensures responses adhere to expected HTTP status codes.
* **Body Content Validation:** Verifies that response body attributes meet expected values or formats.
* **Dynamic Token Handling:** Uses the {{token}} environment variable for secure and dynamic authentication across requests.

### **Sample Test Script**

For the **Get Organization** request:

pm.test("Status code is 200", function () {

pm.response.to.have.status(200);

});

pm.test("Organization name matches", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.data.name).to.eql("PriceLabs");

});

## **6. Execution Instructions**

1. Import both the collection and environment JSON files into Postman.
2. Set the active environment to pricelabs\_prod.
3. Update the token variable in the environment with a valid JWT token.

## **7. Key Highlights**

* **Scalability:** The collection supports dynamic environments for testing across multiple stages (e.g., dev, prod).
* **Security:** Sensitive data like tokens are stored in environment variables, reducing exposure risks.
* **Automation Ready:** Built-in test scripts enable quick validation of API functionality and facilitate CI/CD integration.

## **8. Future Enhancements**

* Include additional scenarios for edge cases and error handling.
* Automate token generation and refresh.

**Screenshots**:











