DATA SUBSCRIBER API GUIDE V1.0

TABLE OF CONTENTS

ABOUT THIS GUIDE **PAGE 3**

SECTION 1: ENROLLMENT **4**
SECTION 2: REGISTER/LOGIN (AUTHORIZATION/AUTHENTICATION) **5-7**
SECTION 2.1: AUTHENTICATION 5-6
SECTION 2.2: TOKEN 7
SECTION 3: CARD ENROLLMENT **8-9**
SECTION 4: GET TRANSCATIONS **10**
SECTION 5: GET CARD DETAILS **11**
SECTION 6: MERCHANT ENROLLMENT & LOCATION **12-13**
SUBSECTION 6.1: MERCHANT ENROLLMENT 12
SUBSECTION 6.2:ADD MERCHANT LOCATION 13
SECTION 7: GET MERCHANT TRANSACTIONS **14**
SECTION 8: MERCHANT DISCOVER **15**
SECTION 9: BANK ACCOUNT ENROLLMENT **16**

=

About This Guide.

The purpose of this document is to provide any information that you would need to integrate with our API endpoints. If you are not able to find the information you are looking for, or have any questions, please refer to our Help Center. Our Quickstart guide gives you a step by step walk through for your entire PentaData integration process. Please be sure to review our Privacy Policy before using our API production.

Developer Workflow:

REGISTER	/SUBSCRIBER/SIGNUP #POST
LOGIN	/SUBSCRIBER/LOGIN #POST
ENROLL CARD	/CARD #POST
GET CARD DETAILS	/CARD/<ID> #GET
CARD TRANSACTIONS	/CARD/<ID>/TRANSACTIONS #GET
ENROLL MERCHANT	/MERCHANT #POST
CREATE LOCATION	/MERCHANT/LOCATION #POST
MERCHANT TRANSACTIONS	/MERCHANT/<ID>/TRANSACTIONS #GET
MERCHANT DISCOVER	/MERCHANT/DISCOVER? MERCHID=<ID> #GET
ENROLL BANK ACCOUNT	/ACCOUNT #POST

1. ENROLLMENT.

API Keys and Authentication

To gain access to the PentaData API, you will need to create an account on our dashboard.

www.PentaDatainc.com/signup

<ENTER SCREEN SHOT HERE OF SIGNUP PAGE>

Once the signup process is complete, you will be provided with a live client_id, secret and public_key via the dashboard.

2. REGISTER/LOGIN

2.1 Authentication/Authorization

Authentication is needed for requests to Pentadata's API or when utilizing one of our endpoints. Json Web Token (JWT) technology is used to verify your identity (authentication) and to confirm the requested operation (authorization).

* Obtain a JWT by logging in with your email and API-key
*All headers on your future requests must include the JWT

Look up your secret key on your dashboard

<Insert screenshot of dashboard where to access secret key>

The login endpoint is available at

/subscriber/login/

and it accepts POST requests, with application/json content and a payload with fields "email" and "key".

In cUrl:

```
curl -X POST \
ADD-URL-HERE/subscriber/login/ \
-H 'Content-Type: application/json' \
-d '{"email": "ADD-EMAIL", "api_key": "ADD-SECRET-KEY"}'
```

2. REGISTER/LOGIN

In Python:

```
## Change these values
email = "  # Your subscriber email
api_key = 'a***' # Your Pentadata API-KEY
url = ' # Pentadata base URL
##
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer ' + api_key
}
endpoint = '/subscriber/login/'
payload = {
    'email': email,
    'api_key': api_key,
}
response = requests.post(
    url + endpoint,
    headers={'Content-Type': 'application/json'},
    json=payload,
)
```

If your data is validated then Pentadata server will respond with the token and a message, e.g.,

```
{
"error": null,
"token": "eyJ********_f3Qo",
"validity_seconds": 18000
}
```

2. REGISTER/LOGIN

2.2 Token Use

Grab the token from our server and include it in the header of all future requests

Your HTTP headers must be like the following ones.

In cUrl:

```
curl \
URL/ENDPOINT/ \
-H 'Content-Type: application/json' \
-H 'Authorization: Bearer ADD-TOKEN-HERE'
# more headers and request payload
```

In Python

```
headers = {
   'Content-Type': 'application/json',
   'Authorization': f'Bearer {token}',
}
```

From this moment on, all your requests will be authenticated and authorized based on the token in the headers. If the token has expired, the server will tell you with an error message and you can request a new one.

3. CARD ENROLLMENT

Enrolling a card means registering a card's 16-digit number, also known as Primary Account Number (PAN) with our API.

- For security issues, you must enter the PAN on our secure website interface after logging in. You should send the PAN via API as it will be rejected. If your needs are different, please reach out to your onboarding representative.
- The "cardId" is a numeric value, unrelated to the PAN, that will have to used in future to safely refer to this card. We don't store the PAN in our servers. You will be able to look up each "cardId" from your dashboard, along with non-sensitive information (e.g., last four digits).

/card
The request payload must contain:

- The PAN (16 digits), "pan".
- Cardholder's first name, "firstName".
- Cardholder's last name, "lastName".
- Expiration month (2 digits), "expMonth".
- Expiration year (2 digits), "expYear".
- The country code, "countryCode". E.g., "USA".

For example:

In cUrl:

```
curl –X POST \
URL/card/link
# all headers here, with JTW
-d ' {\
```

“pan”:”4444444444441111”,\
“firstName”: “Jane”,\
“lastName”: “Doe”,\
“expMonth”: 11,\
“expYear”:2023,\
“countryCode”: “USA”\
}’

3. CARD ENROLLMENT

In Python

```
endpoint = '/card'
payload = {
   'PAN': 4444444444441111,
   'expMonth': 1,
   'expYear': 2023,
   'firstName': 'Jane',
   'lastName': 'Doe',
   'countryCode': 'USA',
}
headers = {
   'Content-Type': 'application/json',
   'Authorization': f'Bearer {JWT}',
}
resp = requests.post(
   URL + endpoint,
   headers=headers,
   json=payload,
)
```

If the enrollment operation succeeds, then our API will respond with a message like the following

```
{
“error”: null,
“message”: “Card linked”,
“cardId”: 81348
}
```

4. GET TRANSACTIONS

By sending a request to this endpoint, you will be able to retrieve a list of transactions for a given card. (You must have enrolled the same card previously)

The endpoint accepts GET requests and is available at:

/card/<id>/transactions

Where <id> must be a valid “cardId” of a card that was enrolled by you. Requests for cards that weren’t enrolled within your account will be rejected (in other words, you can’t just type random numbers as cardId).

The payload should be empty for these requests but remember to add your JWT in the headers.

For example:

In cUrl:

```
curl \
URL/card/534863/transactions \
-H 'Content-Type: application/json' \
-H 'Authorization: Bearer eyJ********0E'
```

In Phyton:

```
endpoint = f'/card/{card_id}/transactions'
headers = {
```

```
    'Content-Type': 'application/json',
    'Authorization': f'Bearer {JWT}',
}
resp = requests.get(
    URL + endpoint,
    headers=headers,
)
```

If the request is correct, then the server will respond with a list of transactions. Each of them contains fields such as:

- “createdAt”, the timestamp of the transaction.
- “pending”, true or false.
- “amount”.
- “street”, “city”, “country”, “streetNo”, “zip”, of the merchant’s location.
- “merchantName”.

5. GET CARD DETAILS

This endpoint allows you to double check non-sensitive card details. You should use it primarily to make sure that the cardID you have corresponded to the correct card.

The endpoint accepts GET requests and is available at:

/card/<id>/

Here too, the card must have been enrolled by you, and the request you send must contain a valid token.

For Example:

In cUrl:

```
curl \
URL/card/42344/ \
-H 'Content-Type: application/json' \
-H 'Authorization: Bearer eyJ********0E'
```

In case of correct request, the server will respond with non-sensitive card details.

For example:

```
{
“cardId”: 42344,
“lastFour”: 1111,
```

“firstName”: “Jane”,
“lastName”: “Doe”,
“expMonth”: 12,
“expYear”: 2023,
“enrolledAt”: 20200101-0800,
}

6. Merchant Enrollment & Location

6.1 Merchant Enrollment

The merchant enrollment endpoint allows a subscriber to register a merchant. This is needed to look up transactions at your store location. You will need a live account with PentaData and a valid API token (JWT). See Section 1.

To enroll as a merchant, you can query the endpoint

/merchant/

that accepts POST requests.

For example:

In cUrl:

curl –X POST \
URL/merchant/enroll/ \
-H ‘Authorization: Bearer Xx883******q3m9384”

If the request is accepted, our server will respond with a message like the one below. Otherwise, it will send an error message and you should correct your request based on it, before trying again.

{
“error”: null,
“message”: “Merchant enrolled”,
“mechantId”: 849412
}

6. MERCHANT ENROLLMENT & LOCATION

6.2 Add Merchant Location

This step is needed if you need to retrieve transactions at certain merchant locations.

To add a new location, send a POST request to the endpoint at

/merchant/location/

Along with your current api token (JWT) and the following location details:
If you correctly provide the first five, then the last two can be skipped

- Street
- Street Number
- Country
- Zip code
- City
- Latitude
- Longitude

For example:

In cUrl:

```
curl -X POST \
URL/merchant/location \
-H 'Content-Type: application/json' \
-H 'Authorization: Bearer eyJ*********kd4' \
-d '{\
"location": {\
"countryCode": "GBR",\
"street": "Liverpool St",\
"streetNo": "162",\
"city": "London",\
"zip": "AAA 001"\
}\
}'
```

You will receive a message like this from our server to let you know if the location was created successfully:

{
“error”: null,
“message”: “Location created”,
“locationId”: 2384347234
}

7. GET MERCHANT TRANSACTIONS.

This endpoint allows you to get a list of transactions from all your merchant locations

Send a GET request to

/merchant/<id>/transactions/

Where <id> must be your own “merchantId”, that was returned to you when you enrolled as a merchant. You can look it up in your dashboard.

For example:

In cUrl:

curl \
URL/merchant/213/transactions \
-H 'Content-Type: application/json'
-H 'Authorization: Bearer eyJ0*******d0E'

Our server will respond with a list of transactions. Each item in the list will contain fields such as

- “amount”.
- “createdAt”.
- “locationId”.
- “street”, “streetNo”, “city”, “country”, “zip”, referred to the location.
- “lastFour”, referred to the card used in the transaction.

8. MERCHANT DISCOVER

This endpoint allows you to look up a merchant's detail with only their ID on a card statement.

GET requests is available at
/merchant/discover?merchid=<ADD-ID>

For example:

In cUrl:

```
curl \
URL/merchant/discover\?merchId\=DOLIUMPTYLTDWELSHPOOLWA \
-H 'Content-Type: application/json' \
-H 'Authorization: Bearer eyJ0**********Cn246E'
```

And in Python:

```
endpoint = "/merchant/discover"
params = {"merchId": "DOLIUMPTYLTDWELSHPOOLWA"}
headers = {} # set content-type and authorization like above
resp = requests.get(
    url + endpoint,
    headers=headers,
    params=params
)
```

If a match is found for that merchant ID, the response will be like the following one:


```
{
"result": {
   "merchants": [
     {
      "ID": "DOLIUMPTYLTDWELSHPOOLWA",
      "address": "UNIT 2 248 WELSHPOOL RD",
      "category": "5533   AUTOMOTIVE PARTS ACCESSORIES STORES",
      "city": "WELSHPOOL",
      "comment": 100,
      "confScore": 100,
      "countryCode": "AUS",
      "countryName": "AUSTRALIA",
       "dbaName": "DOLIUM PTY LTD",
      "locationId": 344147943,
```

```
      "phoneNumber": 893582575,
      "zip": 6106
    }
  ],
  "message": "Atleast 1 merchant found."
}
```


9. Bank Account Enrollment

This endpoint allows you to enroll an entire bank account within our system.
Your bank sends us a secret token that we can use only for a few operations, such as getting available card, transactions, etc.

The functionality is available at

/account

as a POST endpoint.

In cUrl:

```
curl –X POST\
URL/account \
-H ‘Content-Type: application/json’\
-H ‘Authorization: Bearer ****’\
-d ‘{“bank”: “citi”}’
```

And in Python

```
endpoint = /account
headers = {“Content-Type”: “application/json”, “Authorization”: f”Bearer: {token}”}
payload = {“bank”: “citi”}
resp = requests.post(
    url + endpoint,
    json=payload,
    headers=headers,
)
```

When you send this request our API will reply with a JSON response that contains a secure URL where you (or your application’s logic must log in).

Note that this is a safe URL that redirects you to your bank’s website. When you log in with your username and password, your bank does not share with us this information.

We will only get from your bank a secure token that lets us fetch information from your account, according to the permissions that you give us.