Lüt Turbo Outgoing Payments

API Integration Guide

V0.1 DRAFT

July 10, 2026

Table of Contents

1.0 Introduction

Lüt Turbo is a merchant payments platform designed to streamline outgoing payments, invoice collections, connection management, funding, settlements, and ACH processing in a single operational workflow. It provides merchants with a controlled and auditable way to manage counterparties, initiate transactions, apply risk and AML controls, and track money movement from creation through settlement.

The capabilities of the platform can be accessed using a modern web interface or through an API integration with the merchant's e-commerce platform. This document outlines the processes and APIs necessary to integrate directly with the Turbo outgoing payments platform.

1.1 Preparing for Integration

Before beginning the integration process, Lüt recommends scheduling a session with an Integration Specialist who will:

- Review the workflows and APIs relevant to your specific use case
- Provide access to an integration test environment
  (Please note that production configuration settings and keys are specifically meant for the production environment and cannot be used in the integration (test) environment.)

1.2 Environments

Environment	Base URL
Sandbox/demo	https://turbo-backend.preprod.mylut.com
Production	https://turbo-backend.mylut.com

The sandbox/demo environment is financially isolated and does not perform real-world settlement or move actual funds. Transactions from a test merchant system can be sent to this environment safely, without any impact on live financial activity.

The production environment is a live system, and any transactions sent to it will be processed in the real world and will result in actual financial settlement or movement of funds. Only authorized, production-ready transactions should be directed to this environment.

1.3 Key Features

Integrating with the Lüt Turbo system is achieved using REST APIs secured by an API key. The API key is unique to the merchant and production keys should be secured and never shared. A new key can be generated at any time through the Turbo merchant portal. When a new key is created, the old API key is immediately invalidated and the new key is active.

Any connections or outgoing payments created through the API interface can be immediately viewed in the Turbo merchant portal.

1.4 Examples and Notes

In this document, API examples are provided with curl scripts. Some elements of the script will have to be modified. For example, the URL of the end-point (sandbox vs. production), as well as API keys and other transaction specific elements.

2.0 Outgoing Payment

2.1 Endpoint

POST /createPayment

2.2 Purpose

This API is used to create an outgoing payment using the provided connection id and payment amount.

2.3 Key Steps

- Send the API request to Create Payment endpoint
- Pass the API Key in the header
- Provide the payment amount, Payment Method and Connection ID
- Submit the request

2.4 Curl

```
curl --location 'https://turbo-backend.preprod.mylut.com/createPayment' \
--header 'API-KEY:
08f28cddb3c8647b8bcdb9159dd56c9649db6f280dfaf0abb7b6e3c95257ffe8' \
--header 'Content-Type: application/json' \
--data-raw '{
    "amount": 0.51,
    "paymentMethod": "eonline",
    "connectionId": "59dd56c9649db6f280dfaf-59dd56c9649db6f280dfaf-
59dd56c9649db6f280dfaf"
}'
```

2.5 API Response

1. If API-Key is missing in headers

```
{
"response": 2,
"responseCode": 400,
"responseText": "Bad Request",
"responseObject": null
}
```

2. If API-KEY is invalid


```
{
"response": 2,
"responseCode": 401,
"responseText": "Invalid API-KEY",
"responseObject": null
}
```


3. Any one of these: connection id, amount or paymentMethod missing


```
{
"response": 2,
"responseCode": 211,
"responseText": "Mandatory fields are missing",
"responseObject": null
}
```


4. If connection id does not exist or cannot be used (inactive, account info missing)


```
{
"response": 2,
"responseCode": 415,
"responseText": "Active Connection Not Found",
"responseObject": null
}
```


5. If amount is Zero or Less than Zero


```
{
"response": 2,
"responseCode": 412,
"responseText": "Amount should be greater than zero",
"responseObject": null
}
```


6. If Payment Method is not eonline


```
{
"response": 2,
"responseCode": 413,
"responseText": "Invalid payment method",
"responseObject": null
}
```

7. Transaction processed successfully


```
{
   "response": 1,
   "responseCode": 100,
   "responseText": "Success",
   "responseObject": {
         "transactionId": "ce5f1e34-9230-4ca3-ba2f-a5ec8c587308",
         "createdAt": "2026-04-24T10:38:03.000+00:00",
         "connectionEmail": "johny@gmail.com",
         "amount": 1680.00,
         "status": "PENDING"
    }
}
```

3.0 Create Connection

3.1 Endpoint

POST /merchant/connection

3.2 Purpose

Before a merchant can create an outgoing payment through POST /CreatePayment, the merchant must first have a connection created. A connection represents the beneficiary or counterparty that the merchant may pay or invoice.

3.3 Rules

3.3.1 Request Fields

Field	Required	Validation / Rule
apiKey	Yes	Must identify an active merchant/API configuration.
firstName	Conditional	Mandatory if isBusiness = No.
lastName	Conditional	Mandatory if isBusiness = No.
email	Yes	Must be valid email format and unique for the merchant.
accountNumber	Conditional	Required if any bank account field is provided. Otherwise must be null.
routingNumber	Conditional	Required if any bank account field is provided. Otherwise must be null.
accountName	Conditional	Required if any bank account field is provided. Otherwise must be null.
accountType	Conditional	Required if any bank account field is provided. Must be Checking or Saving. Otherwise must be null.
isBusiness	Yes	Must be Yes or No; must align with merchant B2B/B2C configuration.
businessName	Conditional	Mandatory if isBusiness = Yes.

Field	Required	Validation / Rule
contactName	No	Used only when isBusiness = Yes; 1–40 characters, if provided.
phone	No	Optional. Must be 10 digits, if provided.
addressLine1	No	Optional.
addressLine2	No	Optional.
city	No	Optional.
state	No	Optional. Must be a valid full state/territory name, if provided (see list of 51 states and 4 territories validated in the UI).
zipCode	No	Optional. Must be 5 digits if provided.
country	No	Optional.

3.3.2 Bank Account Field Rule

The bank account fields must be treated as an all-or-nothing group.

The four fields are:

accountNumber
routingNumber
accountName
accountType

Rules:

- If the API provides any one of the four fields, then all four fields are mandatory.
- If the API does not provide bank account information, then all four fields must be null. Partial bank account information is not allowed.
- accountType must be either Checking or Saving.

3.3.3 Duplicate Email Rule

Connection email must be unique.

If the same email is submitted where a connection already exists, the API will return an error response and the existing connection object.

3.4 Curl

```
curl --location 'https://turbo-backend.preprod.mylut.com/merchant/connection' \
--header 'Content-Type: application/json' \
--header 'API-KEY:
82780efa10423257db340896037ac0363de2b633a8738214bbfb98437e7a44fc' \
--data-raw '{
    "email": "vamshi88@mylut.com",
    "isBusiness": true,
    "businessName": "Acme Corporation",
    "contactName": "Jane Smith",
    "firstName":"first name",
    "lastName":"last name",
    "accountNumber": "1111222233331111",
    "routingNumber": "110000000",
    "accountName": "Acme Corporation",
    "accountType": "CHECKING",
    "phoneNumber": "9876543210",
    "addressLine1": "456 Business Ave",
    "addressLine2": "Suite 100",
    "city": "Dallas",
    "state": "Texas",
    "zipCode": "75201",
    "country": "USA"
}'
```

3.5 API Response

Success:

{
"response": 1,
"responseCode": 100,
"responseText": "Success",
"responseObject": {
"connectionId": "ca5de199-9c16-478a-9384-05b7e7d47e22",
"status": "active",
"firstName": "first name",
"lastName": "last name",
"email": "accounst1@acme.com",
"isBusiness": false,
"businessName": null,

```
        "contactName": "Jane Smith",
        "phoneNumber": "9876543210",
        "addressLine1": "456 Business Ave",
        "addressLine2": "Suite 100",
        "city": "Dallas",
        "state": "TX",
        "zipCode": "75201",
        "country": "USA",
        "accountNumber": "1111",
        "routingNumber": "0000",
        "accountName": "Acme Corporation",
        "accountType": "checking",
        "paymentReady": true
        }
}
```


Error:


```
{
"response": 2,
"responseCode": 400,
"responseText": "Bad Request",
"responseObject": null
}
{
"response": 2,
"responseCode": 401,
"responseText": "Invalid API-KEY",
"responseObject": null
}
{
"response": 2,
"responseCode": 316,
"responseText": "Invalid email address",
"responseObject": null
}
{
"response": 2,
"responseCode": 320,
"responseText": "account type must be checking or savings",
"responseObject": null
}
```

{
"response": 2,
"responseCode": 322,
"responseText": "Account number must be 4-17 characters",
"responseObject": null
}
{
"response": 2,
"responseCode": 328,
"responseText": "Routing number must be 9 digits",
"responseObject": null
}
{
"response": 2,
"responseCode": 430,
"responseText": "Please enable Business To Business for this merchant",
"responseObject": null
}
{
"response": 2,
"responseCode": 431,
"responseText": "Please enable Business To Individual for this merchant",
"responseObject": null
}{
"response": 2,
"responseCode": 432,
"responseText": "First Name and Last Name are required for individual account",
"responseObject": null
}
{
"response": 2,
"responseCode": 433,
"responseText": "Business Name is required for business account",
"responseObject": null
}
{
"response": 2,
"responseCode": 434,
"responseText": "Contact Name cannot exceed 40 characters",
"responseObject": null
}

{
"response": 2,
"responseCode": 435,
"responseText": "Phone must contain exactly 10 digits",
"responseObject": null
}
{
"response": 2,
"responseCode": 436,
"responseText": "Zip Code must contain exactly 5 digits",
"responseObject": null
}
{
"response": 2,
"responseCode": 437,
"responseText": "Account Number, Routing Number, Account Name and Account Type must all be provided together",
"responseObject": null
}
{
"response": 2,
"responseCode": 438,
"responseText": "Please provide a valid state",
"responseObject": null
}
{
"response": 2,
"responseCode": 439,
"responseText": "Given email is associated with another connection",
"responseObject": null
}
{
"response": 2,
"responseCode": 310,
"responseText": "Connection Already Exists",
"responseObject": null
}

4.0 Edit Connection

4.1 Endpoints

PUT /merchant/connection

4.2 Purpose

This API is used to update an existing merchant connection in the Lüt Turbo system by passing the connection ID along with all the details of the connection. Note that the information provided overwrites the entire connection information in the system. Specific fields within cannot be changed independently.

4.3 Rules

The rules to edit the connection are the same as those for a new connection. In addition, the connection id (this was returned when the connection was created) must be provided.

There is an additional status field that can be set to active or inactive.

4.4 Curl

```
curl --location 'https://turbo-backend.preprod.mylut.com/merchant/connection' \
--header 'Content-Type: application/json' \
--header 'API-KEY:
82780efa10423257db340896037ac0363de2b633a8738214bbfb98437e7a44fc' \
--data-raw '{

"connectionId": "550e8400-e29b-41d4-a716-446655440000",
    "email": "vamshi88@mylut.com",
    "isBusiness": true,
    "businessName": "Acme Corporation",
    "contactName": "Jane Smith",
    "firstName":"first name",
    "lastName":"last name",
    "accountNumber": "1111222233331111",
    "routingNumber": "110000000",
    "accountName": "Acme Corporation",
    "accountType": "CHECKING",
    "phoneNumber": "9876543210",
    "addressLine1": "456 Business Ave",
    "addressLine2": "Suite 100",
    "city": "Dallas",
    "state": "Texas",
```

```
    "zipCode": "75201",
    "country": "USA",
    "status": "active"
}'
```

4.5 API Response

Success:


```
{
"response": 1,
"responseCode": 100,
"responseText": "Success",
"responseObject": {
        "connectionId": "ca5de199-9c16-478a-9384-05b7e7d47e22",
        "status": "active",
        "firstName": "first name",
        "lastName": "last name",
        "email": "accounst1@acme.com",
        "isBusiness": false,
        "businessName": null,
        "contactName": "Jane Smith",
        "phoneNumber": "9876543210",
        "addressLine1": "456 Business Ave",
        "addressLine2": "Suite 100",
        "city": "Dallas",
        "state": "TX",
        "zipCode": "75201",
        "country": "USA",
        "accountNumber": "1111",
        "routingNumber": "0000",
        "accountName": "Acme Corporation",
        "accountType": "checking",
        "paymentReady": true
        }
}
```


Error Response:


```
{
  "response": 2,
  "responseCode": 401,
  "responseText": "Invalid API-KEY",
```

```
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 439,
  "responseText": "Email is required.",
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 331,
  "responseText": "Connection Id is required.",
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 329,
  "responseText": "Status must be Active or Inactive.",
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 317,
  "responseText": "Invalid account details",
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 320,
  "responseText": "account type must be checking or savings",
```

```
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 328,
  "responseText": "Routing number must be 9 digits",
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 322,
  "responseText": "Account number must be 4-17 characters",
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 435,
  "responseText": "Phone number must contain exactly 10 digits",
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 434,
  "responseText": "Contact Name cannot exceed 40 characters",
  "responseObject": null
}

{
  "response": 2,
  "responseCode": 316,
  "responseText": "Invalid email address",
```

```
  "responseObject": null
}
{
  "response": 2,
  "responseCode": 433,
  "responseText": "Business Name is required for business account",
  "responseObject": null
}
{
"response": 2,
"responseCode": 430,
"responseText": "Please enable Business To Business for this merchant",
"responseObject": null
}
{
"response": 2,
"responseCode": 431,
"responseText": "Please enable Business To Individual for this merchant",
"responseObject": null
}
{
"response": 2,
"responseCode": 432,
"responseText": "First Name and Last Name are required for individual account",
"responseObject": null
}
{
"response": 2,
"responseCode": 433,
"responseText": "Business Name is required for business account",
"responseObject": null
}
```

5.0 Last 5 Outgoing Payments

5.1 Endpoint

GET /recentOutgoingPayments

5.2 Purpose

Merchants can submit outgoing payments through the POST /CreatePayment API. This API is used to fetch the 5 most recent outgoing payments for a merchant from the Lüt Turbo system.

This API will only return payment requests submitted through the API, not payments manually created in the merchant portal.

In case of a non-response to a Create Payment API request, this API can be used by the merchant integrator to confirm that the payment request was received and processed, or not. This will help ensure duplicate payments don't occur.

5.3 Key Steps

The most recent 5 outgoing payment requests submitted by API are returned. Note, there may be less than 5, even zero, payments in this set! Results are sorted by the most recent first. Payment details and current status of the payment is also returned.

5.4 Curl

```
curl --location 'https://turbo-backend.preprod.mylut.com/recentOutgoingPayments' \
--header 'API-KEY:
82780efa10423257db340896037ac0363de2b633a8738214bbfb98437e7a44fc'
```

5.5 API Response

Success:


```
{
  "response": 1,
  "responseCode": 100,
  "responseText": "Success",
  "responseObject": [
    {
```

```
      "transactionId": "99352bc8-1950-4931-8e99-03a2eeffa4da",
      "createdAt": "2026-07-03T08:36:32.000+00:00",
      "connectionEmail": "regtest-customer@lut.com",
      "amount": 0.51,
      "status": "BEING PROCESSED"
    },
    {
      "transactionId": "9c8b7bc6-1e60-4f05-b8fa-0030fe48806f",
      "createdAt": "2026-07-02T15:35:19.000+00:00",
      "connectionEmail": "regtest-customer@lut.com",
      "amount": 0.51,
      "status": "BEING PROCESSED"
    },
    {
      "transactionId": "7ac3f392-ce93-49f0-8b6f-1e1f4ce94684",
      "createdAt": "2026-07-02T11:53:31.000+00:00",
      "connectionEmail": "regtest-customer@lut.com",
      "amount": 0.01,
      "status": "BEING PROCESSED"
    },
    {
      "transactionId": "ce5f1e34-9230-4ca3-ba2f-a5ec8c587308",
      "createdAt": "2026-04-24T10:38:03.000+00:00",
      "connectionEmail": "pavani@gmail.com",
      "amount": 1680.00,
      "status": "BEING PROCESSED"
    },
    {
      "transactionId": "18c70bc9-47ae-4044-bdc3-713e007fd553",
      "createdAt": "2026-04-24T09:56:16.000+00:00",
      "connectionEmail": "pavani@gmail.com",
      "amount": 1680.00,
      "status": "BEING PROCESSED"
    }
  ]
}
```


Errors:

```
{
  "response": 2,
  "responseCode": 401,
  "responseText": "Invalid API-KEY",
  "responseObject": null
}
```

```
{
  "response": 2,
  "responseCode": 400,
  "responseText": "Bad Request",
  "responseObject": null
}
```


**** END OF DOCUMENT ****