

Open API Testing Use Cases

By Category — Payment Link / Pay·Void·Refund / Recurring Payment / Treasury Service

Overview

Category	Positive Flows	Negative / Edge Flows	Total
1. Payment Link	17	21	38
2. Pay / Void / Refund	10	15	25
3. Recurring Payment	9	12	21
4. Treasury Service	9	15	24
Total	45	63	108

1. Payment Link

Scope: Payment link creation and management via [/open_payments](#) endpoint.

Key APIs:

- [POST /svc/payment/api/v1/openapi/open_payments](#) - Create a payment link
- [POST /svc/payment/api/v1/openapi/transactions/check](#) - Check payment link status

1.1 Positive Flows

Case No.	Test Scenario	API	Expected Result
PL-P01	Create simple payment link (minimal fields)	POST /open_payments	Link created, amount and reference number match

Case No.	Test Scenario	API	Expected Result
PL-P02	Create payment link with line items	POST /open_payments	Line items preserved, total matches sum of items
PL-P03	Create payment link connected to a customer	POST /open_payments	Link associated with customer, retrievable via query
PL-P04	Create payment link with due date	POST /open_payments	Due date stored, link valid before due date
PL-P05	Create payment link with tips	POST /open_payments	Total = charge_fee + tips, tips recorded
PL-P06	Create payment link with callback & redirect URLs	POST /open_payments	URLs stored, payment redirects to redirect_url
PL-P07	Create payment link with save_this_card enabled	POST /open_payments	Card tokenization option shown, token saved if opted in
PL-P08	Create payment link with pre-authorization only mode	POST /open_payments	Transaction type = Authorization, allow_capture = true
PL-P09	Create payment link with note	POST /open_payments	Note field preserved in order details
PL-P10	Create payment link with multiple line items (3+)	POST /open_payments	All line items preserved with correct price/quantity/total
PL-P11	Query payment link status after creation	POST /transactions/check	HTTP 200, order details and state returned correctly
PL-P12	Create payment link with different currencies (HKD/USD/JPY)	POST /open_payments	Link created with correct currency, amount precision matches currency
PL-P13	Create payment link with payment_method_option.online.show (e.g. "visa,mastercard")	POST /open_payments	Only Visa and Mastercard shown on payment page, other methods hidden
PL-P14	Create payment link with payment_method_option.online.hide (e.g. "alipay,payme")	POST /open_payments	Alipay and PayMe hidden on payment page, all other methods shown
PL-P15	Create payment link with payment_entry_option.online.show (e.g. "google_pay,apple_pay")	POST /open_payments	Only Google Pay and Apple Pay entry shown, Samsung Pay and manual hidden
PL-P16	Create payment link with payment_entry_option.online.hide (e.g. "samsung_pay")	POST /open_payments	Samsung Pay hidden, all other entry methods shown
PL-P17	Create payment link with single method in show (e.g. "wechat")	POST /open_payments	Only WeChat Pay shown on payment page

1.2 Negative / Edge Flows

Case No.	Test Scenario	API	Expected Result
PL-N01	Negative amount	POST /open_payments	Rejected: invalid amount
PL-N02	Missing required field (reference_number)	POST /open_payments	Validation error: required field missing
PL-N03	Duplicate reference_number	POST /open_payments	Rejected: duplicated reference number
PL-N04	Due date in the past	POST /open_payments	Rejected or immediately expired
PL-N05	Note exceeds 255 characters	POST /open_payments	Validation error: exceeds max length
PL-N06	Invalid callback URL format	POST /open_payments	Validation error: invalid URL format
PL-N07	Line item total mismatch (price×qty ≠ total)	POST /open_payments	Rejected: amount mismatch
PL-N08	Zero amount	POST /open_payments	Rejected or accepted as free link
PL-N09	Minimum positive amount (0.01)	POST /open_payments	Link created, amount correctly recorded
PL-N10	Wrong decimal places for currency	POST /open_payments	Rejected: invalid decimal precision
PL-N11	Non-numeric amount (string)	POST /open_payments	Rejected: invalid amount format
PL-N12	Empty line items array	POST /open_payments	Rejected or accepted with no items
PL-N13	Very large amount (upper boundary)	POST /open_payments	HTTP 200 if within limit, otherwise rejected
PL-N14	Missing charge_fee	POST /open_payments	Validation error: charge_fee is required
PL-N15	Invalid currency code (e.g. XXX)	POST /open_payments	Rejected: unsupported currency
PL-N16	show and hide both set in payment_method_option.online	POST /open_payments	Rejected: show and hide are mutually exclusive
PL-N17	Invalid payment method in show (e.g. "paypal")	POST /open_payments	Rejected: invalid payment method, not in enum list

Case No.	Test Scenario	API	Expected Result
PL-N18	Invalid payment entry in hide (e.g. "stripe")	POST /open_payments	Rejected: invalid entry method, not in enum list
PL-N19	Empty string in show (e.g. "")	POST /open_payments	Rejected or ignored: invalid value
PL-N20	show with trailing comma (e.g. "visa,")	POST /open_payments	Rejected: invalid format, trailing comma
PL-N21	show and hide both set in payment_entry_option.online	POST /open_payments	Rejected: show and hide are mutually exclusive

2. Pay / Void / Refund

Scope: Order payment, void, refund, and capture lifecycle.

Key APIs:

- [POST /svc/payment/api/v1/openapi/orders](#) - Create an order
- [POST /svc/payment/api/v1/openapi/open_payments](#) - Create payment transaction
- [POST /svc/payment/api/v1/openapi/orders/refund](#) - Refund a transaction
- [POST /svc/payment/api/v1/openapi/orders/void](#) - Void an order
- [POST /svc/payment/api/v1/openapi/transactions/void](#) - Void a transaction
- [POST /svc/payment/api/v1/openapi/orders/capture](#) - Capture a transaction

2.1 Positive Flows

Case No.	Test Scenario	API	Expected Result
PVR-P01	Create order and pay successfully	POST /orders → POST /open_payments	Payment success, state = paid
PVR-P02	Full refund a paid transaction	POST /orders/refund	Refund created, state = refunded
PVR-P03	Partial refund a paid transaction	POST /orders/refund	Partial refund, state stays paid

Case No.	Test Scenario	API	Expected Result
PVR-P04	Void a paid order	POST /orders/void	Order reverted, state = unpaid, transaction voided
PVR-P05	Void a specific transaction by UUID	POST /transactions/void	Transaction voided successfully
PVR-P06	Pre-authorize then capture full amount	POST /open_payments → POST /orders/capture	Authorization → Capture, state = paid
PVR-P07	Query order status after payment	POST /orders/check	HTTP 200, order state = paid, transactions returned
PVR-P08	Multiple partial refunds (30 + 30 of 100)	POST /orders/refund × 2	Both refunds succeed, total refunded = 60
PVR-P09	Void then re-pay the same order	POST /orders/void → POST /open_payments	Void succeeds, second payment succeeds, state = paid
PVR-P10	Capture then full refund	POST /orders/capture → POST /orders/refund	Capture success, refund success, state = refunded

2.2 Negative / Edge Flows

Case No.	Test Scenario	API	Expected Result
PVR-N01	Refund amount exceeds transaction amount	POST /orders/refund	Rejected: refund amount exceeds
PVR-N02	Refund a voided transaction	POST /orders/refund	Rejected: cannot refund voided txn
PVR-N03	Void an already voided transaction	POST /transactions/void	Rejected: duplicate void
PVR-N04	Void a refunded transaction	POST /orders/void	Rejected: cannot void refunded txn
PVR-N05	Refund with non-existent transaction UUID	POST /orders/refund	Rejected: transaction not found
PVR-N06	Capture an already captured authorization	POST /orders/capture	Rejected: already captured
PVR-N07	Capture amount exceeds authorization amount	POST /orders/capture	Rejected: capture exceeds auth
PVR-N08	Refund with zero amount	POST /orders/refund	Rejected: invalid amount
PVR-N09	Refund with negative amount	POST /orders/refund	Rejected: invalid amount
PVR-N10	Partial capture (50.00 of 100.00)	POST /orders/capture	Partial capture succeeds

Case No.	Test Scenario	API	Expected Result
PVR-N11	Capture with zero amount	POST /orders/capture	Rejected: invalid amount
PVR-N12	Capture with negative amount	POST /orders/capture	Rejected: invalid amount
PVR-N13	Refund a non-existent order	POST /orders/refund	Rejected: order not found
PVR-N14	Void a non-existent transaction	POST /transactions/void	Rejected: transaction not found
PVR-N15	Refund missing required note field	POST /orders/refund	Validation error: note is required

3. Recurring Payment

Scope: Recurring plan management, billing configs, items, and auto-collection lifecycle.

Key APIs:

- [POST /svc/payment/api/v1/openapi/recurring_payments/plan](#) - Create a plan
- [PUT /svc/payment/api/v1/openapi/recurring_payments/plan/{id}/cancel](#) - Cancel a plan
- [PUT /svc/payment/api/v1/openapi/recurring_payments/plan/{id}/cycles](#) - Add cycles
- [PUT /svc/payment/api/v1/openapi/recurring_payments/plan/{id}/recover](#) - Recover cycle
- [POST /svc/payment/api/v1/openapi/recurring_payments/billing_configs](#) - Create billing config
- [POST /svc/payment/api/v1/openapi/recurring_payments/items](#) - Create an item

3.1 Positive Flows

Case No.	Test Scenario	API	Expected Result
RP-P01	Create recurring plan with auto-collection	POST /recurring_payments/plan	Plan created, status = active
RP-P02	Create billing config	POST /recurring_payments/billing_configs	Billing config created
RP-P03	Create a recurring item	POST /recurring_payments/items	Item created successfully

Case No.	Test Scenario	API	Expected Result
RP-P04	Calculate plan amount	POST /recurring_payments/plan/calculate	Total amount correctly computed
RP-P05	Add cycles to an existing plan	PUT /recurring_payments/plan/{id}/cycles	Cycles added, count updated
RP-P06	Cancel an active plan	PUT /recurring_payments/plan/{id}/cancel	Plan status = cancelled, no further charges
RP-P07	Recover a failed cycle	PUT /recurring_payments/plan/{id}/recover	Cycle recovered, new order created for collection
RP-P08	Get plan orders list	GET /recurring_payments/orders/list	HTTP 200, list of orders with cycle and state info
RP-P09	Edit plan details (name, description)	PUT /recurring_payments/plan/{id}	Plan updated, changes reflected in subsequent queries

3.2 Negative / Edge Flows

Case No.	Test Scenario	API	Expected Result
RP-N01	Create plan with invalid customer UUID	POST /recurring_payments/plan	Rejected: customer not found
RP-N02	Cancel an already cancelled plan	PUT /recurring_payments/plan/{id}/cancel	Rejected: already cancelled
RP-N03	Recover a non-failed cycle	PUT /recurring_payments/plan/{id}/recover	Rejected: invalid state for recovery
RP-N04	Delete a billing config in use	DELETE /recurring_payments/billing_configs/{id}	Rejected: config in use
RP-N05	Payment retry count exceeded	Plan auto-collection	Cycle state = uncollectible after N retries
RP-N06	Grace period expiration after failed collection	Plan auto-collection	Order voided after grace period
RP-N07	Plan with trial period (0.00 first cycle)	POST /recurring_payments/plan	First cycle charged at 0.00
RP-N08	Create plan without name	POST /recurring_payments/plan	Validation error: name is required
RP-N09	Add cycles to a cancelled plan	PUT /recurring_payments/plan/{id}/cycles	Rejected: plan is cancelled
RP-N10	Plan item with zero amount	POST /recurring_payments/items	Rejected: invalid amount
RP-N11	Plan item with negative amount	POST /recurring_payments/items	Rejected: invalid amount

Case No.	Test Scenario	API	Expected Result
RP-N12	Edit a cancelled plan	PUT /recurring_payments/plan/{id}	Rejected: cannot edit cancelled plan

4. Treasury Service

Scope: Payee management, payout requests, ledger accounts, and platform transfers.

Key APIs:

- [POST /svc/ts/api/v1/openapi/payees](#) - Create payee
- [POST /svc/ts/api/v1/openapi/payout_requests](#) - Create payout request
- [GET /svc/ts/api/v1/openapi/payout_requests](#) - List payout requests
- [POST /svc/ts/api/v1/openapi/payout_requests/{id}/cancel](#) - Cancel scheduled payout
- [GET /svc/ts/api/v1/openapi/ledger_accounts](#) - Get ledger accounts
- [POST /svc/ts/api/v1/openapi/platform_transfers](#) - Platform transfer

4.1 Positive Flows

Case No.	Test Scenario	API	Expected Result
TS-P01	Create a payee with bank account	POST /payees	Payee created, status = active
TS-P02	Create a payout request to a payee	POST /payout_requests	Payout created, status = pending
TS-P03	Get ledger accounts balance	GET /ledger_accounts	Account list with balances returned
TS-P04	Create a scheduled payout	POST /payout_requests	Scheduled payout created with schedule_at
TS-P05	Cancel a scheduled payout	POST /payout_requests/{id}/cancel	Payout cancelled successfully
TS-P06	Get payee details by ID	GET /payees/{id}	HTTP 200, payee info with bank account and status
TS-P07	List payout requests with status filter	GET /payout_requests	Filtered list returned, only matching status shown

Case No.	Test Scenario	API	Expected Result
TS-P08	Platform transfer between accounts	POST /platform_transfers	Transfer created, balances updated on both accounts
TS-P09	Delete a payee	DELETE /payees/{id}	Payee deleted, no longer retrievable

4.2 Negative / Edge Flows

Case No.	Test Scenario	API	Expected Result
TS-N01	Invalid reference ID (starts with /)	POST /payout_requests	Error 820009: Invalid reference ID
TS-N02	Invalid payee ID	POST /payout_requests	Error 820001: Invalid payee
TS-N03	Zero or negative amount	POST /payout_requests	Error 820004: Invalid amount
TS-N04	Currency mismatch with payee account	POST /payout_requests	Error 820001: Currency mismatch
TS-N05	Cancel a processing payout	POST /payout_requests/{id}/cancel	Rejected: cannot cancel in current state
TS-N06	Amount below minimum limit	POST /payout_requests	Error 820005: Below minimum
TS-N07	Amount above maximum limit	POST /payout_requests	Error 820006: Above maximum
TS-N08	Wrong decimal places for currency	POST /payout_requests	Error 820010: Invalid decimal places
TS-N09	Payout exceeding available balance	POST /payout_requests	Rejected: insufficient balance
TS-N10	Duplicate reference_id (idempotency)	POST /payout_requests	Same result returned or duplicate error
TS-N11	Cancel an already cancelled payout	POST /payout_requests/{id}/cancel	Rejected: already cancelled
TS-N12	Create payee with invalid bank account number	POST /payees	Rejected: invalid bank account format
TS-N13	Payout equal to available balance (edge)	POST /payout_requests	Success if sufficient, balance = 0.00 after
TS-N14	Platform transfer with zero amount	POST /platform_transfers	Rejected: invalid amount
TS-N15	Get payee with non-existent ID	GET /payees/{id}	Rejected: payee not found

Appendix: Error Code Reference (Treasury)

Error Code	Scenario	Source
820001	Invalid payee (not found, currency mismatch)	Payout
820002	Invalid schedule time	Payout
820003	Invalid currency	Payout
820004	Invalid amount (zero or negative)	Payout
820005	Amount below minimum limit	Payout
820006	Amount above maximum limit	Payout
820008	Invalid remark	Payout
820009	Invalid reference ID (format, starts/ends with /, contains //)	Payout
820010	Invalid decimal places for currency	Payout
500104	System service busy (lock contention)	Payout