A-IT Internet Acquiring: Description of the Interaction Procedure

1. What It Is

This service is designed for accepting payments and is intended for use in e-commerce. Merchants operating in retail trade and the provision of services gain the ability to efficiently automate their payment acceptance process. Our clients can use not only standard payment cards, but also the digital wallet services Masterpass and VISA, as well as pay using QR codes.

We ensure compliance with all security standards and adherence to all rules and requirements of the international payment systems, as well as current Ukrainian legislation in the field of e-commerce. Our payment service uses the API+Frame method of interaction.

2. Algorithm of Work

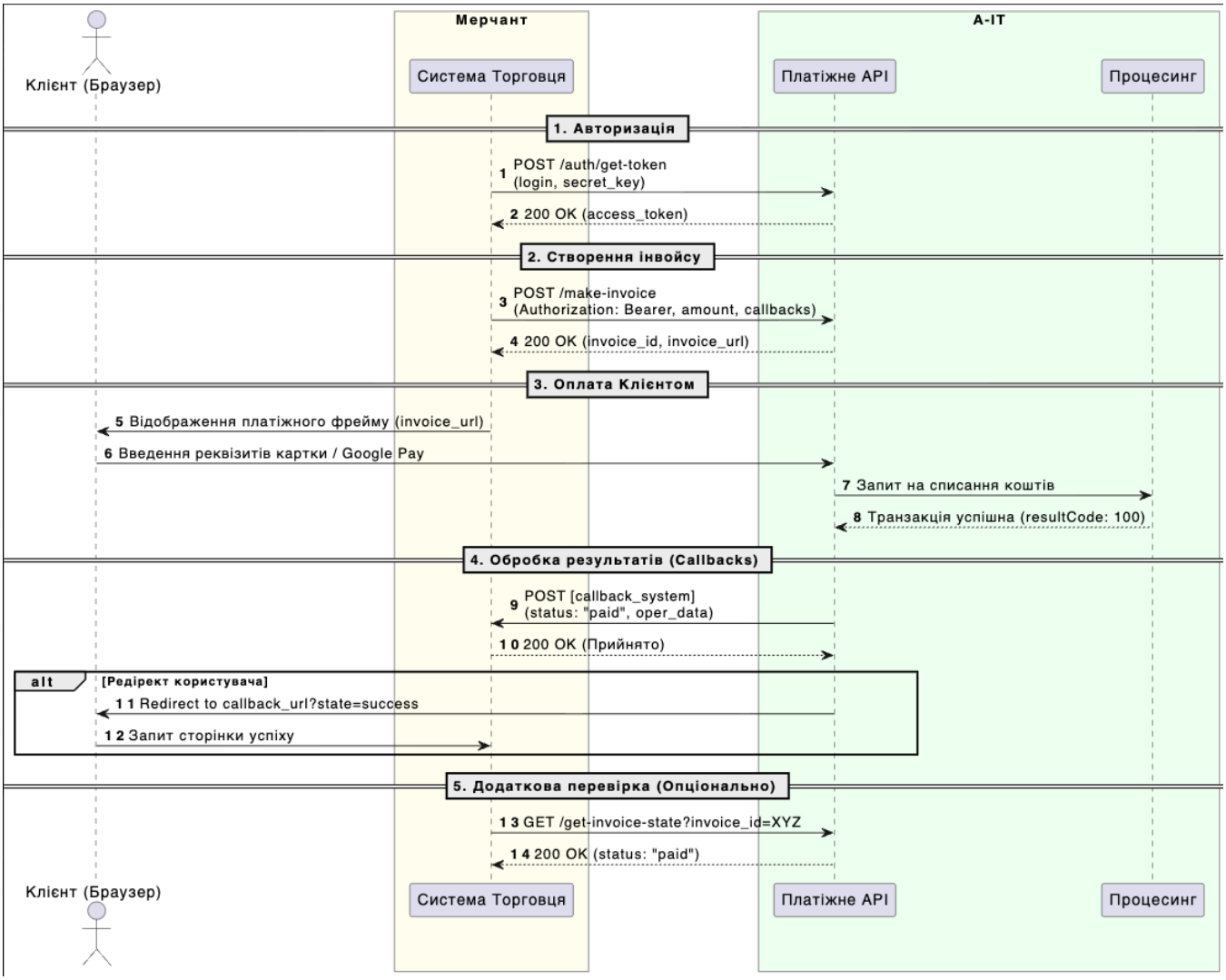


Figure 1. Sequence diagram — general payment algorithm (Authorization, Invoice Creation, Customer Payment, Callback processing, Optional status check)

3. How to Start Testing

To begin testing, you need to implement the integration of the services on the client side. Data exchange is carried out in JSON format using the HTTPS protocol. The client is provided with a key (secret_key) used to generate the digital signature of requests.

Test and Production Environment Parameters

- **login**: <provided by A-IT>
- **secret_key**: <provided by A-IT>

> ***Important:*** *callback_url is the client's URL to which the user is redirected after the operation has been completed. Along with the redirect, a response describing the result of the operation may be passed in the URL parameters and should be analyzed by the merchant. To obtain detailed information about the operation, perform request 4.3 Status Check. To carry out transactions in the test environment, the test card details provided by A-IT must be used.*

4. API Description

4.1. Authorization

- Request format: POST
- Response format: JSON
- URL path: {base_url}/api/v1/internet-acquiring/auth/get-token
- Headers:
    - `Content-Type: application/json`
    - `Accept: application/json`

Request Parameters

Name	Description	Format	Required	Example
`login`	User login	string	M	`testShop`
`secret_key`	Unique key (obtained upon registration)	string	M	`cb92c646-5a2a-4878-90b8-37ca402d2bc5`

Example Request (JSON):

```
{
  "login": "testShop",
  "secret_key": "29741defeb61cb95a056b2bff98c41fb"
}
```

Response Parameters

Name	Description	Format	Example
`access_token`	Authorization token	string	`eyJ0eXAiOiJKV1QiLCJhbGciOiJI`
`token_type`	Token type	string	`bearer`
`expires_in`	Token lifetime (in seconds)	int	`60`
`timestamp`	Date and time of the request	date	`2022-12-14 11:24:12`
`result`	Execution result	string	`ok`
`duration`	Service processing time	float	`0.48`

Example Responsei (JSON):

```
{
   "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.
eyJpc3MiOiJodHRwOi8vcXItYWNxdWlyaW5nLXJlbGVhc2UuYS1iYW5rLmNvbS51YS9hcGkvdjEvaW50ZXJuZXQtYWNxdWlyaW5nL2F1dGgvZ2V0
LXRva2VuIiwiaWF0IjoxNzc5MzUxOTI5LCJleHAiOjE3Nzk5NTY3MjksIm5iZiI6MTc3OTM1MTkyOSwianRpIjoiRFhhMnFLUHZ0TmM1Y3M3NyIs
InN1YiI6IjI2MDUyMWFiYjExaTIyOGV5dHpsIiwicHJ2IjoiY2JjMmExYzljODY5NjA2OTA0OTEyMDRjZDViYTZjZDg1NDg3MDUyZSIsInBheWxv
YWQiOnsiZWtiX2lkIjo3MDcyNTY2OCwiY29tbXVuaWNhdGlvbl91dWlkIjoiYmY4MTcyOTctZDJiYi00ZWU0LTlmNDUtNjJmNzY0ZjM3M2I0Iiwi
Y2hhbm5lbCI6IklOVEVSTkVUIiwibGFuZyI6InVrIn19.h4KGQzvx5pKAy2PzAcIRyhCRuQu-t-_AU6S1rZLOK1I",
   "token_type": "bearer",
   "expires_in": 10080,
   "request_ref": "",
   "response_ref": "6a0ec17909bbb",
   "timestamp": "2026-05-21 11:25:29",
   "result": "ok",
   "duration": 0.85
}
```


4.2. Invoice Creation

- Request format: POST
- Response format: JSON
- URL path: {base_url}/api/v1/internet-acquiring/make-invoice
- Headers:
    - `Content-Type: application/json`
    - `Accept: application/json`
    - `Authorization: Bearer {{access_token}}`

Request Parameters

Name	Description	Format	Required	Example
`amount`	Transaction amount	float	M	`10`
`communication_uuid`	Legacy communication identifier	uuid	Deprecated	`dabfa3dd-ee3f-4d89-a38f262f9d1`
`lifetime`	Invoice lifetime in seconds	int	O	`900`
`description`	Payment purpose (displayed in the frame)	string	M	
`callback_url`	Link for redirecting the client after payment	string	M	`https://host/callback_url`
`callback_system`	Link to which the payment result is sent asynchronously	string	M	`[https://your.domen/callback](https://your.domen/callback)`
`items[]`	Array of information about the items in the invoice	array	M	—
`items.message`	Name of the product or service	string	M	
`items.amount`	Amount per unit of product	float	M	`10`
`items.count`	Quantity	int	M	`1`
`payload{}`	Object with arbitrary merchant data	object	O	—
`order_id_system`	Invoice identifier (order number)	string	O	`test1234`

Example Request (JSON):

```
{
  "amount": "222",
  "communication_uuid": "dabfa3dd-ee3f-4d89-a38f-e60fd262f9d1",
  "items": [
    {
      "message": "Goods services description",
      "amount": 222,
      "count": 1
    }
  ],
  "callback_url": "https://dev-mp.dontburn.space/events/ecg-31_03-453",
  "callback_system": "https://dev-mp.dontburn.space/payments/callback",
  "payload": {
    "key_event_title": "",
    "key_user_name": ""
  },
  "order_id_system": "test1234"
}
```

Callback Processing Procedure

1. callback_system: Sent server-to-server after payment. If your server does not return a successful response, the system retries after: 3, 9, 27, 81, 243, 729, 2187, 6561, 19683, 59049 seconds (up to a total of 24 hours 36 minutes).
2. callback_url: A web page to which the client is redirected, with a status parameter, e.g.: https://host/callback_url?state=success.

Example callback_system Request (JSON):

```
{
  "payload": {
    "testkey": "testValue",
    "testKey2": 111,
    "test_key_3": false
  },
  "name": "Yuryk",
  "address": "",
  "status": "paid",
  "resultCode": "100",
  "resultDesc": "Successful",
  "oper_data": [
    {
      "MERCHANT": "AE0000",
      "CREATEDATE": "22.05.2023 18:37:10",
      "TYPE": "Frame Purchase",
      "REVERSED": "0",
      "REV_AMOUNT": "0",
      "AMOUNT": "10.00",
      "PAN": "537523xxxx6071",
      "APPROVAL": "295180",
      "IDPROCESSING": "6608011",
      "RRN": "314215151925",
      "TRANID": "652d0ebf-b1c8-463b-9310-41238b9ffd3d",
      "ORDER_STATE": "2"
    }
  ]
}
```

Response Parameters

Назва	Опис	Формат	Приклад
`invoice_id`	Unique invoice identifier	string/int	`1477`
`invoice_url`	Link to the payment frame page	string	`https://qr-acquiring...`
`invoice_created`	Invoice creation time	string	`20:50:56`
`expired_at`	ate and time until which the invoice remains active	timestamp	`2024-05-30 10:37:24`
`amount`	Invoice amount	float	`1.00`
`result`	Execution result	string	`ok`

Example Successful Response (JSON):

```
{
    "invoice_id": 1477,
    "invoice_url": "https://base-url/api/v1/clients/payment/8c49179c-4888-40ec-a701-74cbdf3f50dc",
    "invoice_created": "20:50:56",
    "amount": "1.00",
    "message": [],
    "request_ref": "c154886_n62bd979501d14c101a105c56_t639b6c8f1285961204ded512",
    "response_ref": "639b6c8f6a737",
    "timestamp": "2022-12-15 20:50:56",
    "result": "ok",
    "duration": 1.282
}
```

Example Error Response (Limit Exceeded) (JSON):

```
{
  "request_ref": "",
  "response_ref": "6839b59e34f48",
  "timestamp": "2025-05-30 16:41:50",
  "result": "error",
  "duration": 0.487,
  "error": {
    "code": "E0011",
    "message": "    . ,  ,     ."
  }
}
```

4.3. Invoice Status Check

- Request format: GET
- Response format: JSON
- URL path: {base_url}/api/v1/internet-acquiring/get-invoice-state
- Headers:
    - `Authorization: Bearer {{access_token}}`

Вхідні параметри (Request)

Назва	Опис	Формат	Обов'язковість	Приклад
`invoice_id`	Unique invoice identifier	string	M	14

Вихідні параметри (Response)

Назва	Опис	Формат	Приклад
`status`	Transaction status: `new`, `paid`, `cancel`, `failed`, `refund`	string	`paid`
`oper_data.MERCHANT`	Merchant identifier	string	`AE0009`
`oper_data.AMOUNT`	Transaction amount	float	`55`
`oper_data.PAN`	Masked card number (6 + 4)	string	`526961xxxx6263`
`oper_data.TRANID`	Bank transaction identifier (required for reversal	string	`041243219521`

Example Response (JSON):


```
{
  "result": "ok",
  "name": "Yura",
  "address": ". ",
  "status": "cancel",
  "resultCode": "100",
  "resultDesc": "Successful",
  "oper_data": {
    "MERCHANT": "000000020000001",
    "CREATEDATE": "06.07.2022 19:11:01",
    "TYPE": "Pending invoice",
    "REVERSED": "0",
    "REV_AMOUNT": "0",
    "AMOUNT": "105.00",
    "CURRENCY": "980",
    "PAN": "",
    "TRANID": "cd259925-820f-4648-befd-58b5291e4165",
    "ORDER_STATE": "1"
  },
  "response_ref": "62d125c343e22",
  "timestamp": "2022-07-15 11:30:59"
}
```


4.4.Cancellation and Partial Refund

There is no automatic reversal. Refunds are processed via the API from 1 to 180 calendar days from the moment of payment.

- Request format: POST
- URL path: {base_url}/api/v1/internet-acquiring/refund-invoice

Request Parameters

Name	Description	Format	Required	Example
`tran_id`	Transaction ID (oper_data.TRANID)	string	M	`6a16ca2e-5a93-4b08-a971-46b5ce184f24`
`amount`	Refund amount (provided only for a partial refund)	number	O	`25`

Example Full Refund Request (JSON):


```
{
    "tran_id": "6a16ca2e-5a93-4b08-a971-46b5ce184f24"
}
```


Example Response (JSON):

```
{
    "result": "ok",
    "request_ref": "",
    "response_ref": "636e7ca504017",
    "timestamp": "2022-11-11 18:47:35",
    "duration": 0.48
}
```


5. Response Codes (resultCode)

Code	Description	
(resultCode)	**(resultDesc)**	
100	Transaction completed successfully	Successful
101	Transaction completed successfully for a partial amount.	Partially successful
102	Successful, purchase only.	Successful, purchase only.
200	External decline. Call to issuer.	External decline. Call to issuer.
201	Lost or stolen card.	Lost or stolen card.
202	Usage limit exceeded. Call to issuer.	Uses limit exceeded. Call to issuer.
203	Invalid card.	Invalid card
204	Restricted or invalid card status. Call to issuer.	Restricted or invalid card status. Call to issuer.
205	Unable to authorize. Call to issuer.	Unable to authorize. Call to issuer.
206	Insufficient funds.	Insufficient funds.
207	Bad CVV.	Bad CVV
208	Bad CAVV.	Bad CAVV
209	Expired card.	Expired card
210	Ineligible account or account not found.	Ineligible account or Account not found
211	Unable to cancel. Invoice already paid.	Unable to cancel. Invoice already paid
212	Pending invoice.	Pending invoice
213	Masterpass unavailable. Try again later.	Masterpass unavailable. Try again later.
301	Transaction not supported.	Transaction not supported
302	Session not found or inactive.	Session not found or inactive
303	Original transaction not found.	Original transaction not found
304	Ineligible transaction or invalid transaction code.	Ineligible transaction or Invalid transaction code
305	Format error.	Format error
306	Invalid amount.	Invalid amount
307	Personal information input error.	Personal information input error
308	Duplicate transaction.	Duplicate transaction
309	Acquirer limit exceeded. Call to acquirer.	Acquirer limit exceeded. Call to acquirer.
310	Timeout.	Timeout
311	Invalid terminal identifier.	Invalid terminal identifier
312	Card not supported.	Card not supported

313	Transaction not processed.	Transaction not processed
314	CVV input required.	CVV input required
315	Installment confirmation required.	Installment confirm required.
316	Session expired.	Session expired
400	3DS verification required.	3DS Verify required
401	Lookup verification required.	Lookup Verify required
402	Credit limit exceeded.	Credit limit exceeded
403	Invalid OTP.	Invalid OTP
404	Token not found.	Token not found
405	OTP attempts limit exceeded.	OTP tries limit was exceeded
406	Transaction is processing. Try to request status later.	Transaction is processing. Try to request status later
407	Client verification unsuccessful.	Client verification unsuccess
408	Credentials not found.	Credentials not found
409	Restricted operation type. Contact support.	Restricted operation type
410	Country not found.	Country not found
411	Country is prohibited.	Country is prohibited
412	Currency not found.	Currency not found
413	Currency is prohibited.	Currency is prohibited
414	Exchange rate not found.	Exchange rate not found
500	System error. Call to support.	System error. Call to support.
501	Terminal temporarily blocked.	Terminal temporary blocked.
502	Unauthorized usage.	Unauthorized usage
900	Epic fail. Call to support.	Epic Fail. Call to support.

6. Google Pay™ через A-IT Hosted Checkout

Documentation for merchants on using Google Pay™ through the A-IT Hosted Checkout payment platform.

6.1 General Information

Google Pay™ is a modern method of online payment that lets buyers pay for goods and services using payment cards saved in their Google Account.

The A-IT Hosted Checkout solution provides Google Pay™ support without requiring the merchant to integrate directly with the Google Pay API.

6.2 Advantages of Google Pay™

- Fast checkout.
- No need to manually enter card details.
- Tokenization of payment data.
- High level of security.
- Improved payment conversion.

6.3 Integration Model

Google Pay™ is provided through A-IT Hosted Checkout.

The merchant integrates only with the A-IT payment platform API.

Not required:
• a separate integration with the Google Pay API;
• Google Pay certificates;
• processing or decrypting payment tokens;
• separate registration with Google as a Google Pay Merchant.

6.4 Activation for Merchants

Google Pay™ is automatically activated for all merchants using A-IT Hosted Checkout.
Not required:
• submitting a separate application to enable Google Pay™;
• performing additional configuration in the merchant portal;
• integrating with the Google Pay API.

Once the merchant is activated on the platform, Google Pay™ is automatically displayed on the payment page for supported devices and browsers.

6.5 Merchant Requirements

To use Google Pay™, the merchant must:
• be connected to the A-IT payment platform;
• use Hosted Checkout;
• comply with Google's policies - Google Pay™ Acceptable Use Policy
• comply with the Google Pay™ Brand Guidelines- Google Pay™ Brand Guidelines.

6.6 Payment flow

1. The buyer places an order.
2. The merchant creates a payment session via the API.
3. The buyer is redirected to Hosted Checkout.
4. The system checks Google Pay™ availability.
5. The Google Pay™ button is displayed.
6. The buyer selects a card and confirms the payment.
7. Google returns an encrypted payment token.
8. ABank processes the token and performs authorization.
9. The buyer receives the payment result.
10. The merchant receives the callback and can check the status via the API.

6.7 Security

Google Pay™ uses tokenization of payment data.
he actual card details are never passed to the merchant directly. Payment tokens are used to process the transaction, which increases the level of protection for users' financial information.

6.8 Supported Authentication Scenarios

CRYPTOGRAM_3DS — a tokenized transaction with a device cryptogram.
PAN_ONLY — a card transaction without a device cryptogram.

If required, the issuing bank may initiate an additional check via 3D Secure.

6.9 PSD2 та Strong Customer Authentication (SCA)

Google Pay™ supports modern authentication mechanisms and complies with PSD2 requirements.
If required by the issuing bank, Strong Customer Authentication (SCA) and 3D Secure procedures may be applied during the payment.

6.10 Google Pay™ Acceptable Use Policy

All merchants using Google Pay™ must comply with the Google Pay and Wallet APIs Acceptable Use Policy.

Official document:
Google Pay and Wallet APIs Acceptable Use Policy

6.11 Google Pay™ Terms of Service

By using Google Pay™, the merchant confirms acceptance of and compliance with the Google Pay API Terms of Service.

Official document:
Google Pay API Terms of Service

6.12 Google Pay™ Brand Guidelines

Merchants must use the Google Pay™ logos, buttons, and brand elements in accordance with Google's official guidelines.

Official document:
Google Pay™ Brand Guidelines

6.13 Pre-Launch Checklist

Before launch, it is recommended to verify:

- Correct creation of the payment session.
- Display of the Google Pay™ button.
- Successful completion of test payments.
- Processing of callback notifications.
- Retrieval of payment statuses via the API.

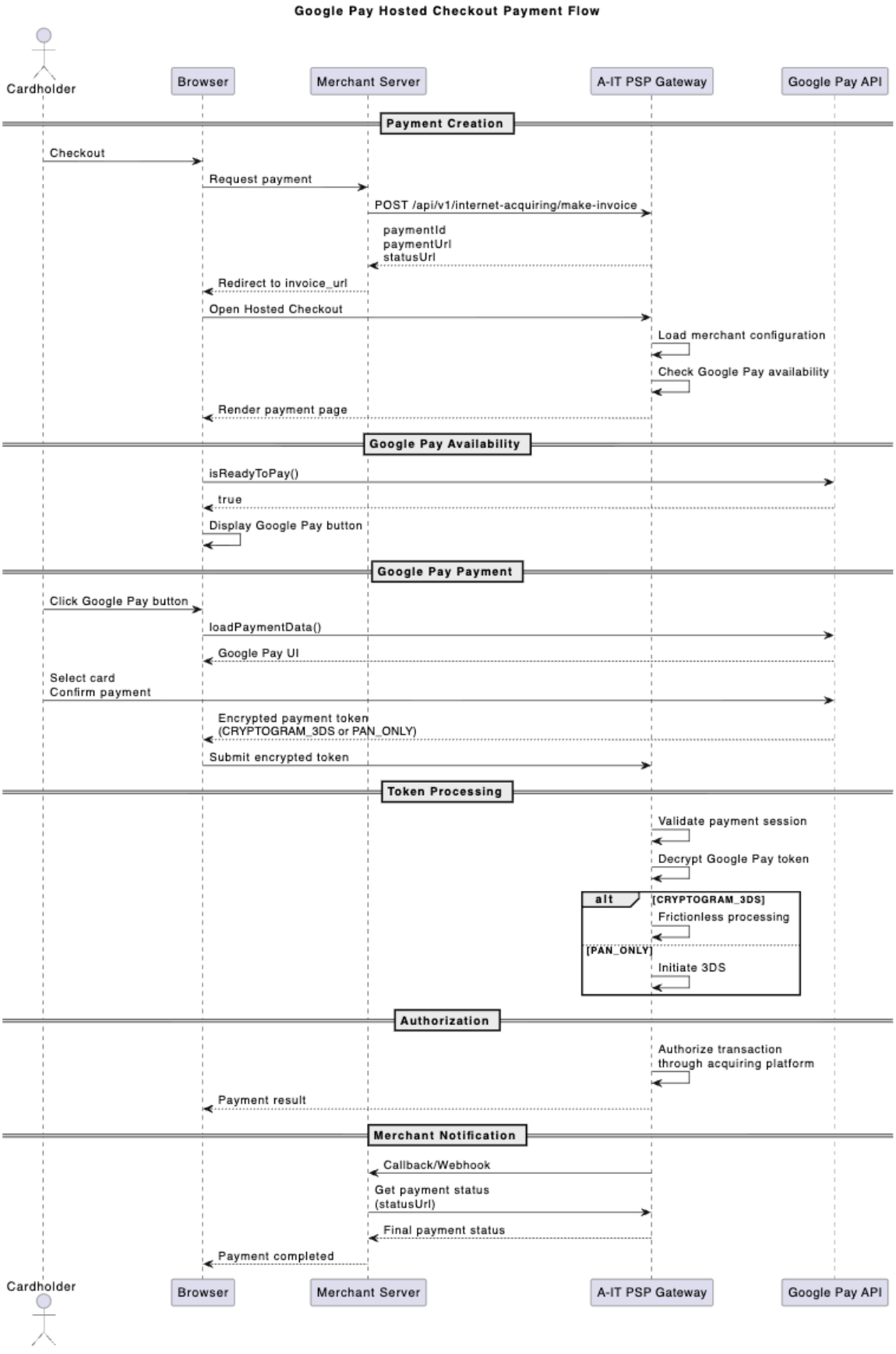
Official checklist:
Google Pay API Integration Checklist

6.14 Developer Documentation

Google Pay API:
Google Pay API Documentation

Google Pay Web:
Google Pay Web Documentation

6.15 UML та Hosted Checkout Flow


Google Pay Hosted Checkout Payment Flow
Cardholder
Browser
Merchant Server
A-IT PSP Gateway
Google Pay API
Payment Creation
Checkout
Request payment
POST /api/v1/internet-acquiring/make-invoice
paymentId
paymentUrl
statusUrl
Redirect to invoice_url
Open Hosted Checkout
Load merchant configuration
Check Google Pay availability
Render payment page
Google Pay Availability
isReadyToPay()
true
Display Google Pay button
Google Pay Payment
Click Google Pay button
loadPaymentData()
Google Pay UI
Select card
Confirm payment
Encrypted payment token
(CRYPTOGRAM_3DS or PAN_ONLY)
Submit encrypted token
Token Processing
Validate payment session
Decrypt Google Pay token
alt
[CRYPTOGRAM_3DS]
Frictionless processing
[PAN_ONLY]
Initiate 3DS
Authorization
Authorize transaction
through acquiring platform
Payment result
Merchant Notification
Callback/Webhook
Get payment status
(statusUrl)
Final payment status
Payment completed
Cardholder
Browser
Merchant Server
A-IT PSP Gateway
Google Pay API