CONFIDENTIAL

TNG CONSOLIDATED ACCOUNT BASED TRANSACTION IMPLEMENTATION (PARKING)

API Specifications

Date   : 04th October 2023
Version   : 0.8

TABLE OF CONTENTS

1 Document Control and Versioning

No	Description	Version	Date	Status	Author
1	1st Draft	0.1	18/3/2019	Draft	ANPD
2	• Remove clientSecret from request header • Updated 3.2.1 diagram • Change 4.2 function description	0.2	29/4/2019	Draft	ANPD
3	• Updated serialNum description • Updated cardNo sample data	0.3	8/5/2019	Draft	ANPD
4	• Updated diagram 3.2.1 and 3.2.2 • Merged 5.1 Get Card Status and 5.2 Forward Parking Entry to 5.1 Get Device Status • Added new column Action to table 5.3.3 • Rename the following fields for 5.2 Transaction o surchargeGSTAmt to surchargeTaxAmt o parkingGSTAmt to parkingTaxAmt	0.4	17/5/2019	Draft	ANPD
5	Document format	0.5	21/07/2019	Release	ANPD
6	• Added HTTP methods to each API • Updated 5.2.3 Response Status • Added orderId and acceptedDateTime in Transaction response • Added more commands to 4.5.1 • Added hash algorithm to 4.5.2	0.6	30/08/2019	Release	ANPD
7	• Add lpr detail in section 5.1.1, 5.2.1 • New section at 6.5 for lpr extend info	0.7	20/09/2023	Draft	TNG
8	• Rectify description on response code 998 at section 7 • Update LPR use case diagram • Add cancel entry api at section 5.3	0.8	04/10/2023	Draft	TNG

	• Remove samTerminalId samMachineCode section 6.5				

2 **Document Scope and Intention**

The purpose of this document is to provide API specifications for integrations between TNG and parking operators. The document provides the high-level view of the transmission requirement and the scope of work expected from the vendor. Note that :

- This document describes the transmission via RESTful API for clear understanding.
- This messaging specification is based upon the current production systems. Changes to the production system may impact this specification and to be updated in due process.
- The overall product scope is PayDirect (ABT), LPR, RFID, and Offline Card Purse.
- This document version will only cover the ABT & LPR Content. RFID and Offline Card Purse content will be finalized in the subsequent versions.

3 Overview

3.1 Architecture Design
3.1.1 Pay-Direct
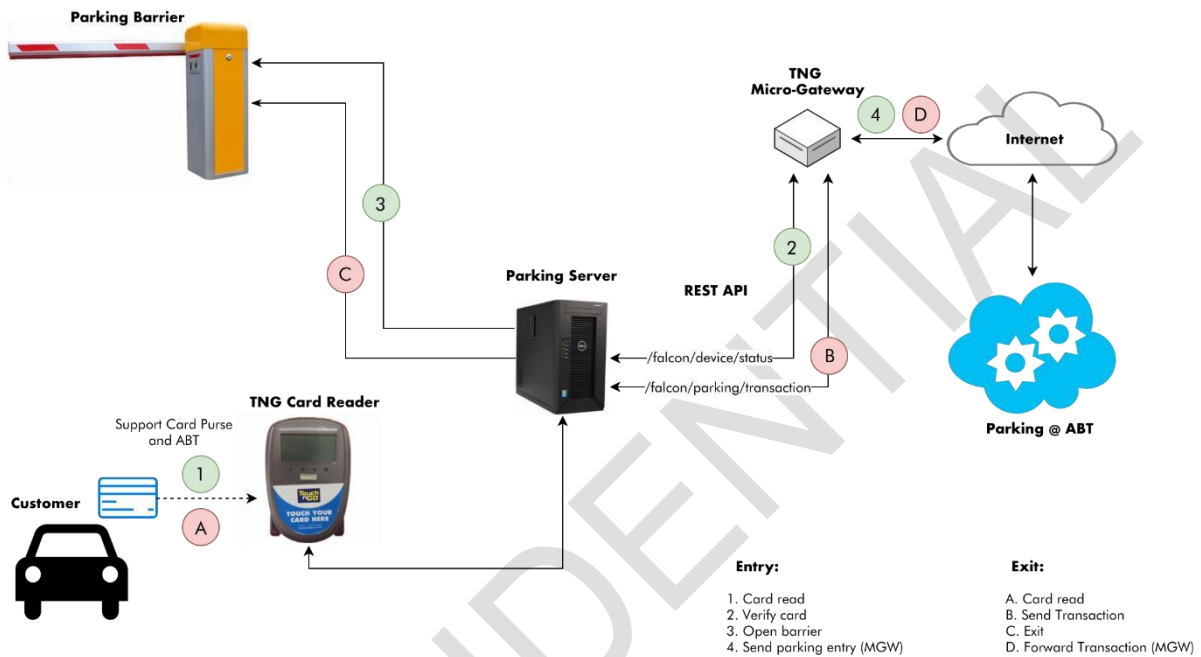
Figure 1

Figure 1 shows the overall architecture for ABT Parking solution. Below is the main system involved:

1) TNG Card Reader
2) Parking Management System (PMS)
3) TNG Micro-Gateway
4) Account Based Transaction System (ABT)

All the data transmissions between PMS and ABT only via TNG Micro-Gateway.

3.1.2 LPR
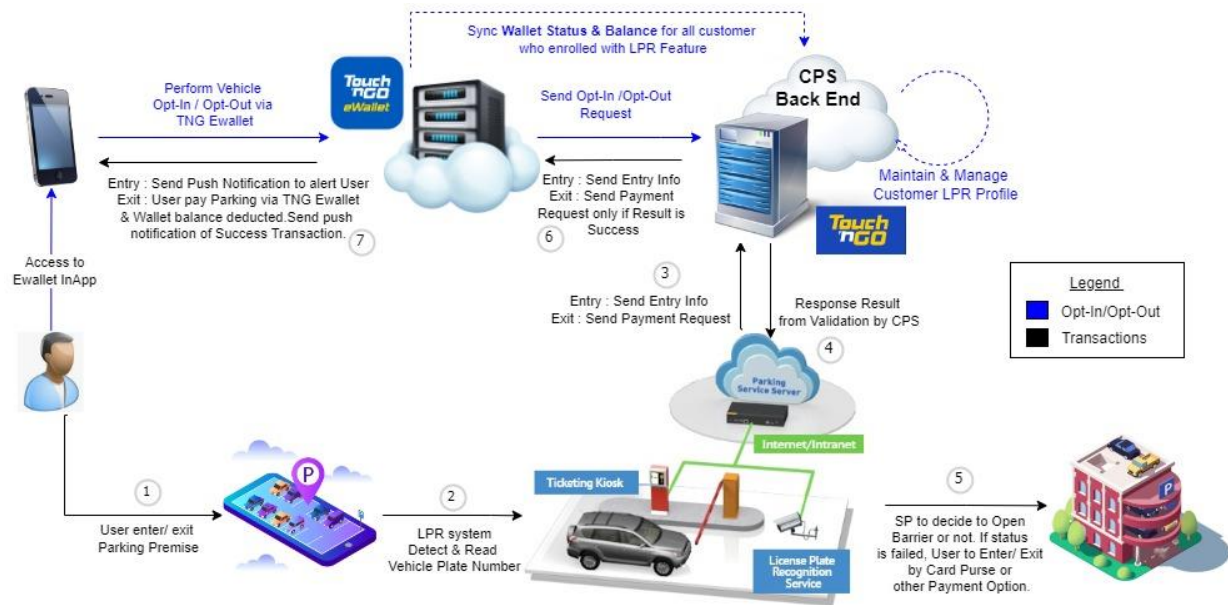
Figure 2

Figure 2 above describes the overall User Journey & Architecture for LPR Parking System. In this system, the main elements involved is as the following :

1) Mobile App - LPR Parking Module
2) License Plate Registration (LPR) reader
3) TNG Card Reader (optional)
4) Parking Management System (PMS)
5) CPS Back-End
6) Payment solution provider (Ewallet)

3.2 Use Cases

The following is the anticipated scenarios during exit with their expected process handling.

| Scenario 1 - SI implements both paydirect and lpr - LPR has precedence. Entry with barrier (for non I City Mall)		
Entry	**Exit**	**Expectation**
LPR ok	LPR ok	Charge via LPR.
LPR ok	LPR fail	Fallback to card only. No paydirect because during entry is LPR.
LPR fail. Enter with paydirect. User tap card during entry.	LPR ok	Charge via paydirect. User tap card during exit.
LPR fail. Enter with paydirect. User tap card during entry.	LPR fail	Charge via paydirect. User tap card during exit.
Scenario 2 - SI implements both paydirect and lpr - LPR has precedence. Entry no barrier.		
Entry	**Exit**	**Expectation**
LPR ok	LPR ok	Charge via LPR.
LPR ok	LPR fail	Fallback to card only. No paydirect because during entry is LPR.
LPR fail.	LPR ok	Need manual intervention, since NEI.
LPR fail.	LPR fail	Need manual intervention, since NEI.
PayDirect ok/fail	-	No applicable since no barrier to force customer tap card during entry
Scenario 3 - SI implements only lpr. Entry with barrier.		
Entry	**Exit**	**Expectation**
LPR ok	LPR ok	Charge via LPR.
LPR ok	LPR fail	Fallback to card purse.
LPR fail.	LPR ok	Need manual intervention, entry gate not open.
LPR fail.	LPR fail	Need manual intervention, entry gate not open.
Scenario 4 - SI implements only lpr. Entry without barrier. (Same as Scenario 2)		
Entry	**Exit**	**Expectation**
LPR ok	LPR ok	Charge via LPR.
LPR ok	LPR fail	Fallback to card purse.
LPR fail.	LPR ok	Need manual intervention, since NEI.
LPR fail.	LPR fail	Need manual intervention, since NEI.
Scenario 5 - SI implement only paydirect.		

Entry	Exit	Expectation
paydirect ok	paydirect ok	Charge via Paydirect.
paydirect ok	paydirect fail	Fallback to card purse.
paydirect fail	paydirect ok	Fallback to card purse.
paydirect fail	paydirect fail	Fallback to card purse.

Scenario 6 - SI implement only paydirect. But entry using LPR. i.e. I City Mall		
Entry	**Exit**	**Expectation**
LPR ok	paydirect ok	Charge via Paydirect. Do entry and exit api calls during exit.
LPR ok	paydirect fail	Fallback to card purse.
LPR fail	paydirect ok	Need manual intervention, since NEI.
LPR fail	paydirect fail	Need manual intervention, since NEI.

Scenario 7 - SI implement both LPR & paydirect. But entry using LPR with barrier (for I City Mall)		
Entry	**Exit**	**Expectation**
LPR ok	LPR ok and paydirect ok	Charge via LPR, if LPR payment fail, charge via paydirect, if Paydirect fail fallback to card purse
LPR ok	LPR ok and paydirect fail	Charge via LPR, if LPR payment fail, Need manual intervention, since NEI.
LPR fail	paydirect ok	Charge via paydirect, if paydirect fail fallback to card purse
LPR fail	paydirect fail	Need manual intervention, since NEI.

3.2.1 Pay with Pay-Direct when LPR payment failure (scenario 7)
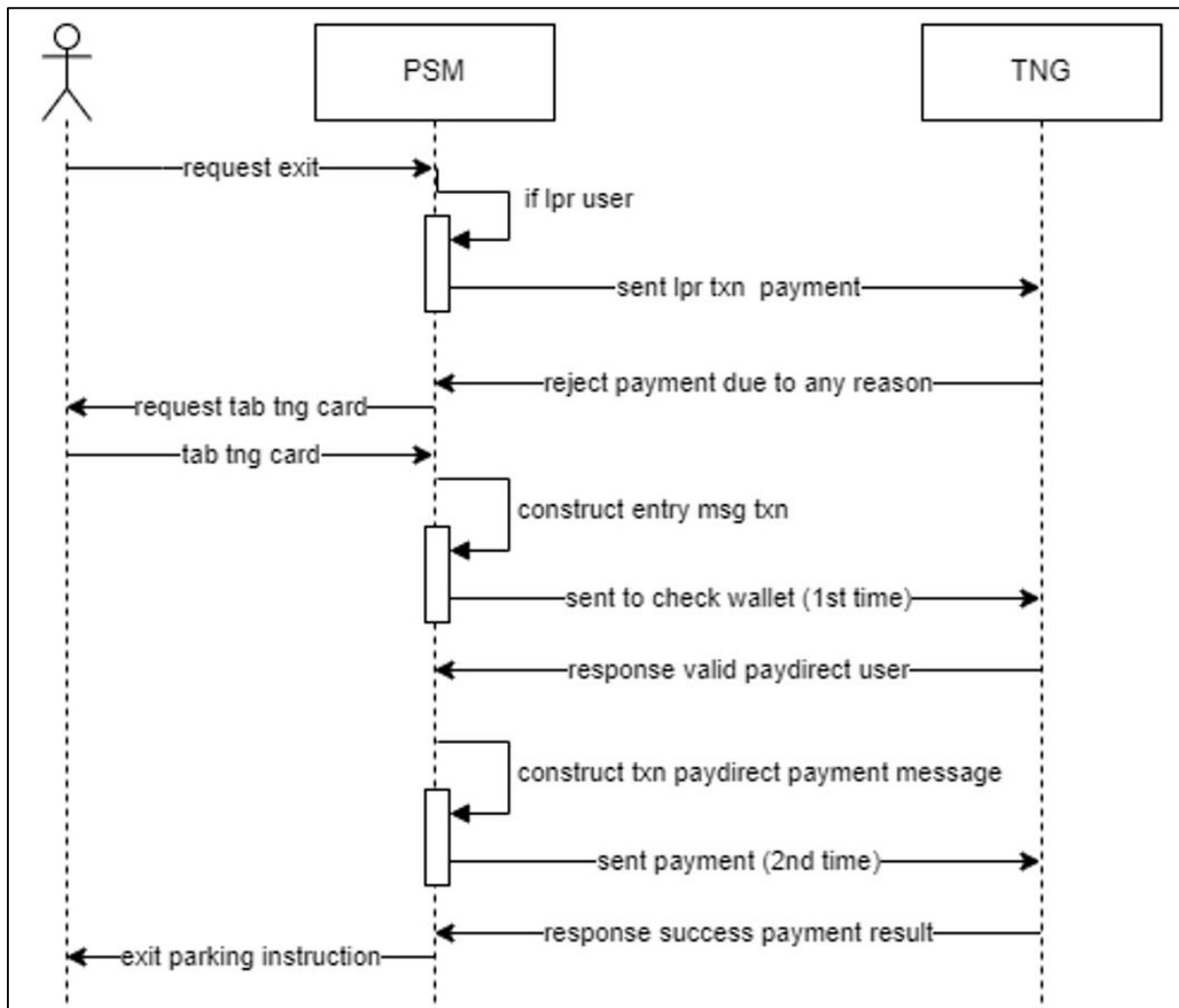
Figure 3

3.2.2   Pay-Direct Parking Enter
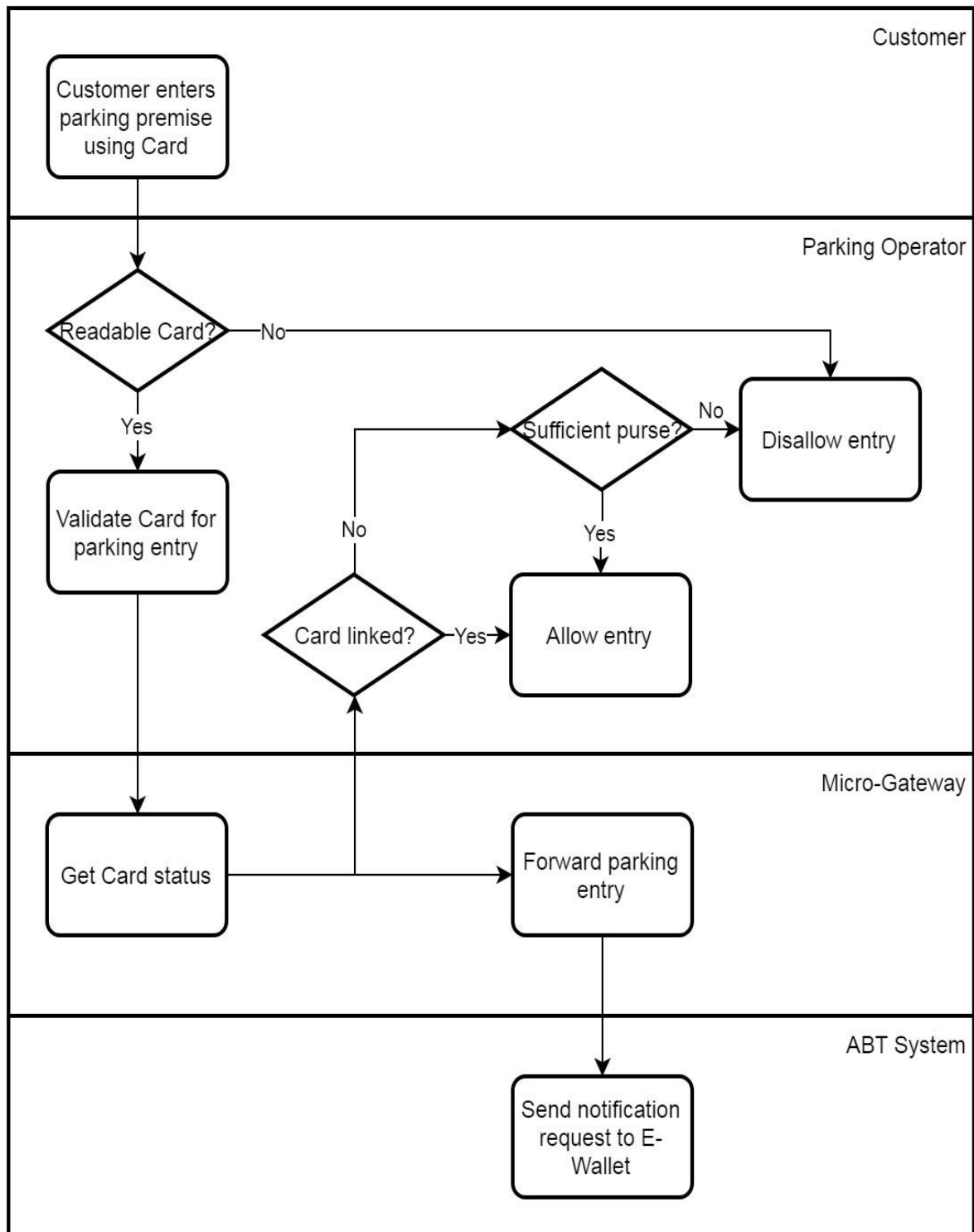
Figure 4

3.2.3   Pay-Direct Parking Exit
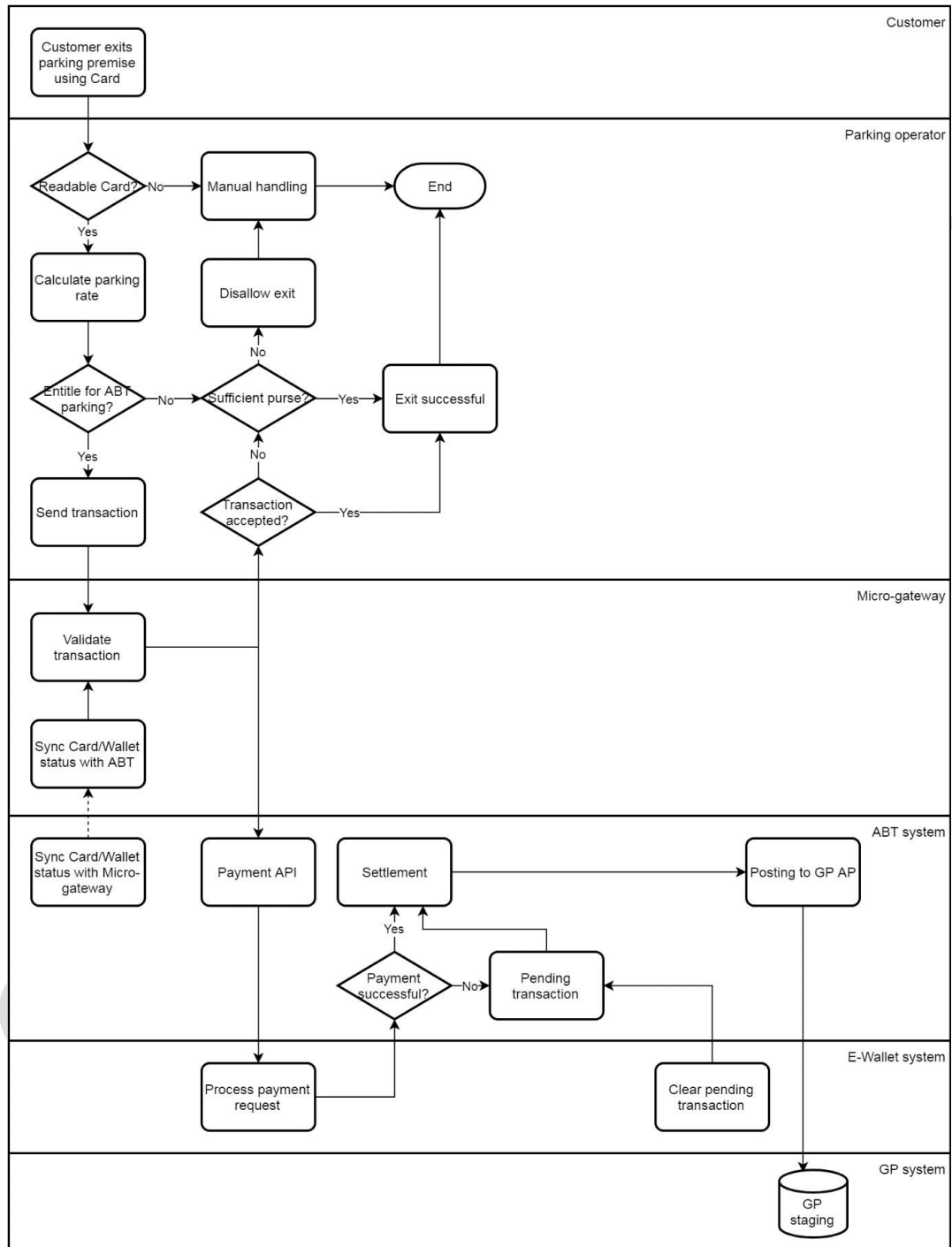
Figure 5

3.2.4 LPR Parking Enter
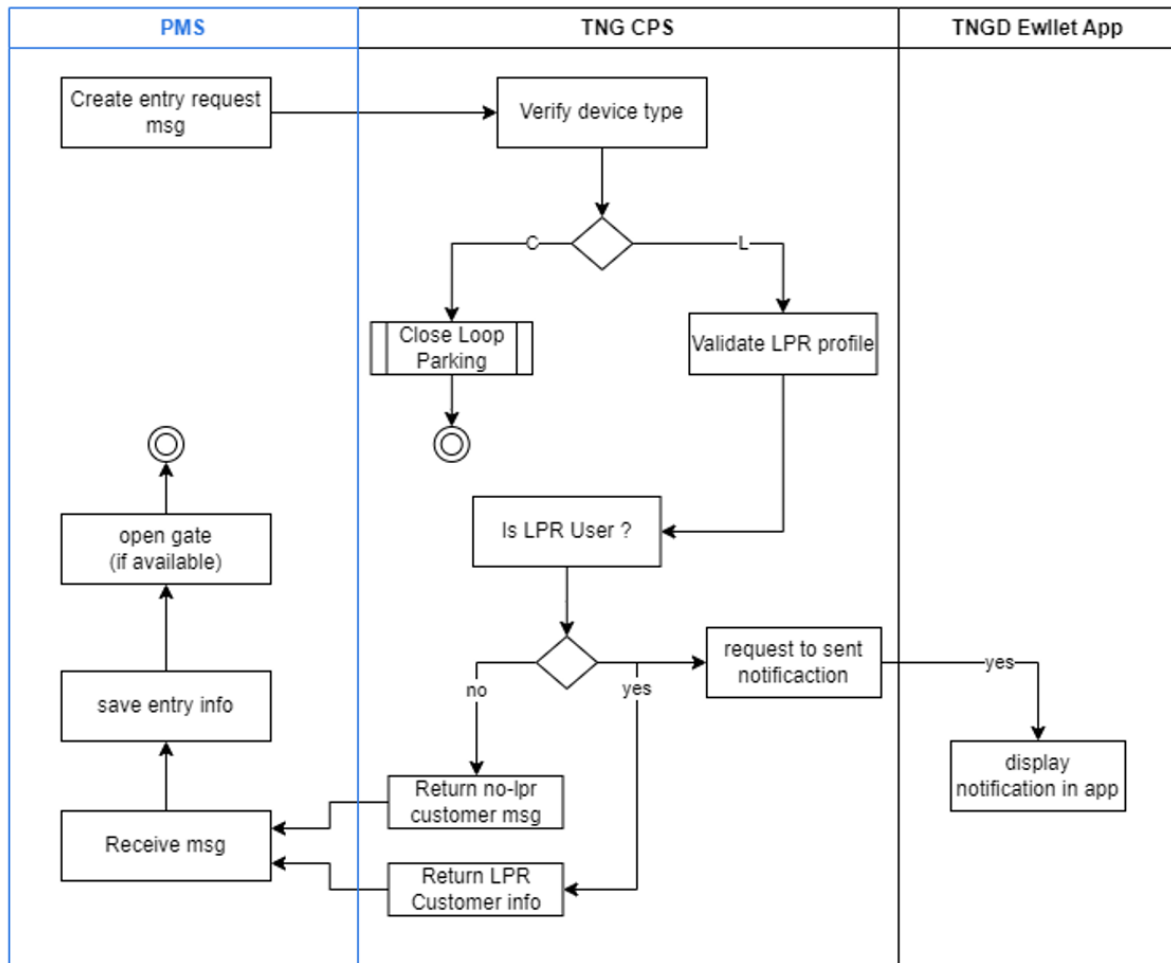
Figure 6

3.2.5 LPR Parking Exit
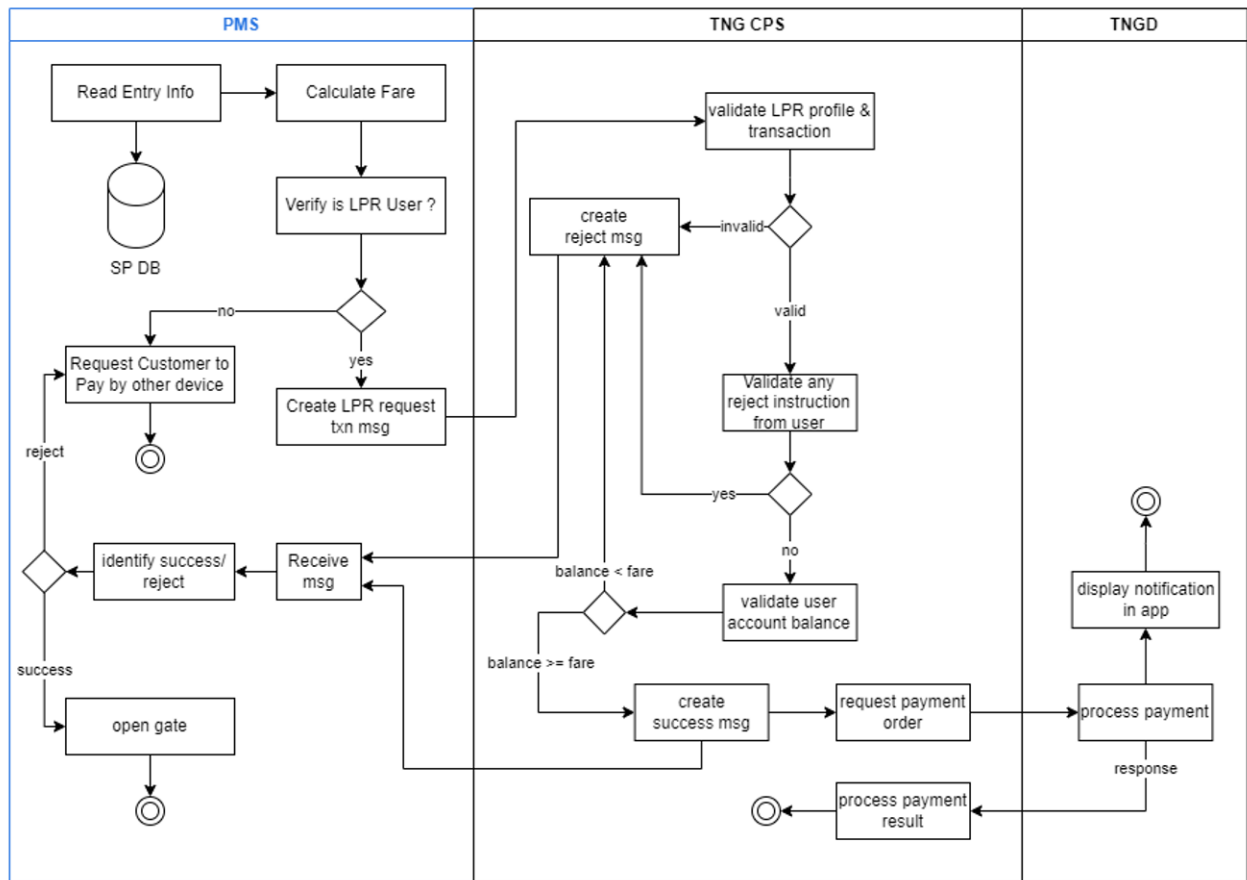
Figure 7

4 **API Structure**

All the APIs provided by TNG Micro-Gateway are RESTful service.

The data transmissions are done on ASCII based messages and JSON string format.

4.1   **Request Structure**

```
{
  "request": {
    "header": {},
    "body": {}
  },
  "signature": ""
}
```

4.2   **Request Header**

No	Parameter	Data Type	Size	M/O	Description
1	requestId	Alphanumeric	64	M	Unique Serial # Alphanumeric ACSII Request ID
2	timestamp	Alphanumeric	-	M	The timestamp of request sent ISO 8601 standard with timezone e.g 2001-07-04T12:08:56.111+05:30

No	Parameter	Data Type	Size	M/O	Description
3	clientId	Alphanumeric	32	M	Provided by ABT, to identify the source.
4	function	Alphanumeric	128	M	Refer to each API Function
5	version	Alphanumeric	8	M	1.0
6	reserve	Alphanumeric	256	O	Additional information. Reserved for future used. {"key":"value"}

4.3 Request Body

The request body structure is defined at each API level.

4.4 Response Structure

4.4.1 Response Header

No	Parameter	Data Type	Size	M/O	Description
1	requestId	Alphanumeric	64	M	Unique Serial # Alphanumeric ACSII Request ID
2	timestamp	Alphanumeric	-	M	The timestamp of response sent ISO 8601 standard with timezone e.g 2001-07-04T12:08:56.111+05:30
3	clientId	Alphanumeric	32	M	Same as Request
4	function	Alphanumeric	128	M	Same as Request
5	version	Alphanumeric	8	M	1.0
6	reserve	Alphanumeric	256	O	Additional information. Reserved for future used. {"key":"value"}

4.4.2 Response Body

The response body structure is defined at each API level.

4.5 Signature

The signature is generated by the value of request and response in the JSON structure. It is recommended to sign and verify with the plaintext, no whitespace and comments.

4.5.1 Secret Key

The secret key (public & private key pair) should be generated by **RSA_2048** algorithm and in Public-Key Cryptography Standards (PKCS) #8 format. The public key needs to be exchanged to ensure signature signed by the private key can be verified by the public key.

Sample commands to generate the key sets:

ssh-keygen -b 2048 -t rsa ssh-keygen -e -f .ssh/id_rsa.pub -m

PKCS8 > .ssh/id_rsa.pub.PKCS8 openssl pkcs8 -topk8 -in .ssh/id_rsa -

out .ssh/id_rsa.pri.PKCS8 -nocrypt

4.5.2 Sign

The parameters in ["request":] should be to be digitally signed and encode the signature using Base64.

When sending the API request with the below JSON structure, it's called **original data**.

Pay-Direct

```json
{
    "header": {
        "clientId": "211020000000000000044",
        "function": "abt.parking.transaction",
        "requestId":
"UmVTJt8cFIJItfUGGQqHlKMVNgaVLBNw2c162ME4llPzVglmekwuiRVeHdtwiLYsi",
        "timestamp": "2017-03-30T05:16:26.000+08:00",
        "reserve": "{}",
        "version": "1.0"
    },
    "body": {
        "cardMfgNo": "100000009"
    }
}
```

LPR

```json
{
    "header": {
        "clientId": "211020000000000000044",
        "function": "abt.parking.transaction",
        "requestId":
"UmVTJt8cFIJItfUGGQqHlKMVNgaVLBNw2c162ME4llPzVglmekwuiRVeHdtwiLYsi",
        "timestamp": "2017-03-30T05:16:26.000+08:00",
        "reserve": "{}",
```

```
        "version": "1.0"

    },

    "body": {

        "vehiclePlateNo": "WWW1234"

    }

}
```

The signature should be added to final request as shown below. Now it's called as **final data**.

The signature string is generated using <u>original data without any changes</u> in the hash algorithm **SHA-256** with the private key and then encode with **Base64**.

The **original data** should be treated as the sub-node of "request" which is at the same level as "signature" node in the final data.

Assemble nodes of "request" and "signature" in JSON structure.

Pay-Direct

```
{

  "request": {

    "header": {

      "clientId": "211020000000000000044",

      "function": "abt.parking.transaction",

      "requestId": "UmVTJt8cFIJItfUGQqHlKMVNgaVLBNw2c162ME4llPzVgImekwuiRVeHdtwiLYsi",

      "timestamp": "2017-03-30T05:16:26.000+08:00",

      "reserve": "{}",

      "version": "1.0"

    },

    "body": {

      "cardMfgNo": "100000009"

    }

  },

  "signature":
```

"MF5oFs9ugR4IUa7RVu4SqnxbIMZsVhyUEDTYxuLGxklJYDzDiuY3y+oCYgPyac9iQ6/xL/gMPtSOD RdG2V/g0G0mrw5DMM6bmG7TPGKyeZG1JW7aUP1GZXWAJis5AGt9BSKyLcfl6o1GXA45hS41DR Jur2y2ZrRXsBEiBtlx2Z/9Asuem0l6JTw4k0+CJw5k/pBCeBU9dcqztOXtLkGh3Z7TvGveHSoz8xeuHM UAyxQb417kmX5B1IHjRcLzX0h6VmnXCdvzEdJL90AhcqUOZeHB+mjZYJC695GKGltmRgqq9FyR F1 txoidk+e2ZV7c/KWhxcFwfREXqsF+19QjzFQ=="

}

LPR

{

  "request": {

    "header": {

      "clientId": "211020000000000000044",

      "function": "abt.parking.transaction",

      "requestId": "UmVTJt8cFIJltfUGQqHIKMVNgaVLBNw2c162ME4llPzVgImekwuiRVeHdtwiLYsi",

      "timestamp": "2017-03-30T05:16:26.000+08:00",

      "reserve": "{}",

      "version": "1.0"

    },

    "body": {

      "vehiclePlateNo": "100000009"

    }

  },

  "signature": "MF5oFs9ugR4IUa7RVu4SqnxbIMZsVhyUEDTYxuLGxklJYDzDiuY3y+oCYgPyac9iQ6/xL/gMPtSOD RdG2V/g0G0mrw5DMM6bmG7TPGKyeZG1JW7aUP1GZXWAJis5AGt9BSKyLcfl6o1GXA45hS41DR Jur2y2ZrRXsBEiBtlx2Z/9Asuem0l6JTw4k0+CJw5k/pBCeBU9dcqztOXtLkGh3Z7TvGveHSoz8xeuHM UAyxQb417kmX5B1IHjRcLzX0h6VmnXCdvzEdJL90AhcqUOZeHB+mjZYJC695GKGltmRgqq9FyR F1 txoidk+e2ZV7c/KWhxcFwfREXqsF+19QjzFQ=="

}

4.5.3   Verification

When to receive payload like the final data above, the steps below need to follow:

1   Split the "request" and "signature" objects.

2   Read request/response body without any changes like **original data**.

3   Decrypt signature string with public key which is pair of private key used by sender to encrypt signature.

4   Verify the correctness of the signature. If correct, consume the original data.

5   API

5.1   Get Device Status

The purpose of this API to verify the device (card/RFID tag) is linked or valid.

Function: falcon.device.status
Version: 1.0
Endpoint: POST https://<base_url>/falcon/device/status

5.1.1   Request Body

No	Parameter	Data Type	Size	M/O	Description
1	deviceInfo	DeviceInfo	-	M	Device info
2	entryTimestamp	DateTime	-	M	The entry timestamp ISO 8601 standard with timezone e.g 2001-07-04T12:08:56.111+05:30

No	Parameter	Data Type	Size	M/O	Description
3	entrySPId	Alphanumeric	2	M	Service Provider ID of entry location
4	entryPlazaId	Alphanumeric	3	M	Location ID of entry location
5	entryLaneId	Alphanumeric	3	M	Lane ID of the entry location
6	extendInfo	String	4096	O	Extend info in JSON string. For deviceType `C`, refer to Card Transaction Extend Info For deviceType `D` refer to RFID Transaction Extend Info For deviceType `L` refer to LPR Transaction Extend Info

Pay-Direct

```
{
    "request": {
        "header": {},
        "body": {
            "deviceInfo": {
                "deviceType": "C",
                "deviceNo": "1000000009"
            },
            "entryTimestamp": "2001-07-04T12:08:56.111+05:30",
            "entrySPId": "01",
            "entryPlazaId": "100",
            "entryLaneId": "100",
            "extendInfo": "{\"cardTran\": \"00001\",\"cardNo\": \"12345678\",\"cardPurse\":
                        5.50,\"cardExpiry\": \"2016-03-03T00:00:00.000+08:00\",\"samTerminalId\":
                        \"12345678\",\"samMachineCode\": \"12345678\"}"
```

```
        }
    },
    "signature": ""
}
```

LPR

```
{
    "request": {
        "header": {},
        "body": {
            "deviceInfo": {
                "deviceType": "L",
                "deviceNo": "WWW1234"
            },
            "entryTimestamp": "2001-07-04T12:08:56.111+05:30",
            "entrySPId": "01",
            "entryPlazaId": "100",
            "entryLaneId": "100",
            "extendInfo": "{\"vehicleModel\":\"TOYOTACAMRY\",\"vehicleColor\":\"WHITE\",
                        \"vehicleType\":\"Motorcar\",\"}"
        }
    },
    "signature": "string signature"
}
```

5.1.2   Response Body

No	Parameter	Data Type	Size	M/O	Description
1	responseInfo	ResponseInfo		M	Response info
2	deviceStatus	Alphanumeric	50	O	The status of the card/tag: • ACTIVE • INACTIVE

```json
{
    "response": {
        "header": {},
        "body": {
            "responseInfo": {
                "responseStatus": "S",
                "responseCode": "000",
                "responseMessage": "Success"
            },
            "deviceStatus": "ACTIVE"
        }
    },
    "signature": " string signature"
}
```

5.1.3   Response Status

Response Status	Response Code	Response Message
F	101	Invalid card/tag/Car Plate No

5.2 Transaction

The purpose of this API to send transaction to ABT upon exit the parking premise.

Function: falcon.parking.transaction

Version: 1.0

Endpoint: POST https://<base_url>/falcon/parking/transaction

5.2.1 Request Body

No	Parameter	Data Type	Size	M/O	Description
1	deviceInfo	DeviceInfo	-	M	Device info
2	serialNum	Alphanumeric	25	M	Alpha-numeric code generated by SP which uniquely identifies each transaction (Any SP unique transaction number for the SP to be able to reconcile back to their system) If transaction is handled separately by multiple SI in the same SP, then transaction serial number to follow according to SI This is a 25-character alphanumeric code, where: a. The first character denotes the product: 0 = RFID 1 = ABT 2 = Parking (ABT/RFID) 3 = LPR b. The next 2 characters signifies SPID. c. The next 3 characters signifies PlazaID. d. The next 3 characters signifies LaneID. e. The next 15 characters are referring to the date and time generated by the PMS. (YYYYMMDDHHMMSSS) f. The last character is reserved. Default as 0.

No	Parameter	Data Type	Size	M/O	Description
					E.g. "282603W072017010112000000 g. 0"
3	transactionType	Alphanumeric	2	M	C = Complete (Closed System – populate the Entry and Exit information)
4	entryTimestamp	DateTime	-	M	The entry timestamp in ISO 8601 standard with timezone e.g 2001-07-04T12:08:56.111+05:30
5	entrySPId	Alphanumeric	2	M	Service Provider ID of exit location
6	entryPlazaId	Alphanumeric	3	M	Location ID of exit location
7	entryLaneId	Alphanumeric	3	M	Lane ID of the exit location
8	appSector	Alphanumeric	3	M	Defaults to 09 (Parking)
9	exitTimestamp	DateTime	-	M	The exit timestamp in ISO 8601 standard with timezone e.g 2001-07-04T12:08:56.111+05:30
10	exitSPId	Alphanumeric	2	M	Service Provider ID of exit location

No	Parameter	Data Type	Size	M/O	Description
11	exitPlazaId	Alphanumeric	3	M	Location ID of exit location
12	exitLaneId	Alphanumeric	3	M	Lane ID of the exit location
13	vehicleClass	Alphanumeric	2	O	00: Motorcycles (Vehicles with two axles and two wheels) 01: Private Cars (Vehicles with two axles and three or four wheels (excluding taxi and bus)) 02: Vans and other small good vehicles (Vehicles with two axles and six wheels (excluding bus)) 03: Large Trucks (Vehicles with three or more axles (excluding bus)) 04: Taxis 05: Busses
14	tranAmt	Double	-	M	The total fare amount charged to customer
15	surchargeAmt	Double	-	M	Surcharge charged to customer
16	surchargeTaxAmt	Double	-	M	This is the tax charged on the surcharge amount. 2 decimal value.
17	parkingAmt	Double	-	M	Parking Amount. 2 decimal value

No	Parameter	Data Type	Size	M/O	Description
18	parkingTaxAmt	Double	-	M	This is the tax charged on the parking fare amount. 2 decimal value.
19	extendInfo	String	4096	O	Extend info in JSON string. For deviceType `C`, refer to Card Transaction Extend Info For deviceType `D` refer to RFID Transaction Extend Info For deviceType `L` refer to LPR Transaction Extend Info

Pay-Direct

```
{
  "request": {
    "header": {},
    "body": {
      "deviceInfo": {
        "deviceType": "C",
        "deviceNo": "1000000009"
      },
      "serialNum": "282603W072017010112000000",
      "transactionType": "C",
      "entryTimestamp": "2017-03-03T10:02:03.215+08:00",
      "entrySPId": "82",
      "entryPlazaId": "603",
      "entryLaneId": "W07",
      "appSector": "09",
      "exitTimestamp": "2017-03-03T10:02:03.215+08:00",
      "exitSPId": "82",
      "exitPlazaId": "603",
      "exitLaneId": "W07",
      "vehicleClass": "01",
```

```
            "tranAmt": 2.20,
            "surchargeAmt": 0.20,
            "surchargeTaxAmt": 0.00,
            "parkingAmt": 2.00,
            "parkingTaxAmt": 0.00,
            "extendInfo": "{\"cardTran\": \"00001\",\"cardNo\": \"123456789012345678\",
                       \"cardPurse\": 5.50,\"cardExpiry\": \"2016-03-03T00:00:00.000+08:00\",
                       \"samTerminalId\": \"12345678\",\"samMachineCode\": \"12345678\"}"
        }
    },
    "signature": "string signature"
}
```

LPR

```
{
    "request": {
        "header": {},
        "body": {
            "deviceInfo": {
                "deviceType": "L",
                "deviceNo": "WWW1234"
            },
            "serialNum": "382603W072017010112000000",
            "transactionType": "C",
            "entryTimestamp": "2017-03-03T10:02:03.215+08:00",
            "entrySPId": "82",
            "entryPlazaId": "603",
            "entryLaneId": "W07",
            "appSector": "09",
            "exitTimestamp": "2017-03-03T10:02:03.215+08:00",
            "exitSPId": "82",
            "exitPlazaId": "603",
            "exitLaneId": "W07",
            "vehicleClass": "01",
            "tranAmt": 2.20,
            "surchargeAmt": 0.20,
            "surchargeTaxAmt": 0.00,
            "parkingAmt": 2.00,
            "parkingTaxAmt": 0.00,
            "extendInfo": "{\"vehicleModel\":\"TOYOTACAMRY\",\"vehicleColor\":\"WHITE\",
                       \"vehicleType\":\"Motorcar\"}"
        }
```

```
    },
    "signature": "string signature"
}
```

5.2.2   Response Body

No	Parameter	Data Type	Size	M/O	Description
1	responseInfo	ResponseInfo		M	Response info
2	orderId	String	64	O	Order ID
3	acceptedDateTime	DateTime		O	Order Accepted Date Time in ISO 8601 standard with timezone e.g 2001-07-04T12:08:56.111+05:30

```
{
    "response": {
        "header": {},
        "body": {
            "responseInfo": {
                "responseStatus": "A",
                "responseCode": "000",
                "responseMessage": "Accepted"
            },
            "orderId": "21102000000000000044201100100201701011200001",
            "acceptedDateTime": "2001-07-04T12:08:56.111+05:30"
        }
    },
    "signature": "string signature"
}
```

5.2.3 Response Status

Response Status	Response Code	Response Message
F	101	Invalid card/tag/vehicle plate number
F	102	Inactive card/tag/LPR status
F	103	Insufficient Balance
F	104	Invalid wallet status
F	201	Invalid serial number
F	202	Invalid SP ID/Loc ID/T-Type
F	203	Transaction exceed time limit

Note: This response codes are yet to be finalized.

5.3 Cancel Entry

The purpose of this API to send instruction to CPS request to cancel the entry.

Function: falcon.parking.cancel

Version: 1.0

Endpoint: POST https://<base_url>/falcon/parking/cancel

5.3.1   Request Body

No	Parameter	Data Type	Size	M/O	Description
1	deviceInfo	DeviceInfo	-	M	Device info
2	entryTimestamp	DateTime	-	M	The entry timestamp ISO 8601 standard with timezone e.g 2001-07-04T12:08:56.111+05:30
3	entrySPId	Alphanumeric	2	M	Service Provider ID of entry location
4	entryPlazaId	Alphanumeric	3	M	Location ID of entry location
5	entryLaneId	Alphanumeric	3	M	Lane ID of the entry location
6	extendInfo	String	4096	O	Extend info in JSON string. For deviceType `C`, refer to Card Transaction Extend Info For deviceType `D` refer to RFID Transaction Extend Info

No	Parameter	Data Type	Size	M/O	Description
					For deviceType `L` refer to [LPR Transaction Extend Info](#)

Pay-Direct

No applicable

LPR

```
{
   "request": {
      "header": {},
      "body": {
         "deviceInfo": {
            "deviceType": "L",
            "deviceNo": "WWW1234"
         },
         "entryTimestamp": "2001-07-04T12:08:56.111+05:30",
         "entrySPId": "01",
         "entryPlazaId": "100",
         "entryLaneId": "100",
         "extendInfo": "{\"vehicleModel\":\"TOYOTACAMRY\",\"vehicleColor\":\"WHITE\",
                      \"vehicleType\":\"Motorcar\"}"
      }
   },
   "signature": "string signature"
}
```

5.3.2   Response Body

No	Parameter	Data Type	Size	M/O	Description
1	responseInfo	ResponseInfo		M	Response info

No	Parameter	Data Type	Size	M/O	Description
2	deviceStatus	Alphanumeric	50	O	The status of the card/tag: • ACTIVE • INACTIVE

```json
{
    "response": {
        "header": {},
        "body": {
            "responseInfo": {
                "responseStatus": "S",
                "responseCode": "000",
                "responseMessage": "Success"
            },
            "deviceStatus": "ACTIVE"
        }
    },
    "signature": "string signature"
}
```

5.3.3   Response Status

Response Status	Response Code	Response Message
F	101	Invalid card/tag/Car Plate No

6 **Objects**

6.1 **ResponseInfo**

No	Parameter	Data Type	Size	M/O	Description
1	responseStatus	Alphanumeric	2	M	The status of the request: S : Success A : Accepted F : Failure
2	responseCode	Alphanumeric	10	M	The status code of the request
3	responseMessage	Alpanumeric	100	M	The status message of the request

6.2 **DeviceInfo**

No	Parameter	Data Type	Size	M/O	Description
1	deviceType	Alphanumeric	1	M	The device type: C : TNG Card R : RFID Tag L : Vehicle Plate No. (LPR)

No	Parameter	Data Type	Size	M/O	Description
2	deviceNo	Alphanumeric	30	M	The device no can be following: 1. Card Mfg No 2. RFID Tag ID 3. Vehicle Plate No
3	vehiclePlateNum	Alphanumeric	30	O	The vehicle plate number

6.3 Card Transaction Extend Info

No	Parameter	Data Type	Size	M/O	Description
1	cardTran	Alphanumeric	5	M	Read Card Transaction number as it is from card
2	cardNo	Alphanumeric	18	M	Read form the card as it is
3	cardPurse	Double	-	M	Read purse value as it is from Card

No	Parameter	Data Type	Size	M/O	Description
4	cardExpiry	DateTime	-	O	The card expiry date ISO 8601 standard with timezone e.g 2001-07-04T12:08:56.111+05:30
5	samTerminalId	Alphanumeric	8	M	This is the Terminal ID info in the Reader's SAM.
6	samMachineCode	Alphanumeric	8	M	This is the Machine Code info in the Reader's SAM.

6.4 RFID Transaction Extend Info
To be defined in future.

6.5 LPR Transaction Extend Info

No	Parameter	Data Type	Size	M/O	Description
1	vehicleModel	Double	32	O	Read form the vehicle model as it is

No	Parameter	Data Type	Size	M/O	Description
2	vehicleColor	DateTime	32	O	Read form the vehicle color as it is
3	vehicleType	Alphanumeric	32	O	Read form the vehicle type as it is

7 General Response Codes

Response Status	Response Code	Response Message
S	000	Success
A	000	Accepted
F	001	Invalid message due to [X]
F	002	Invalid signature
F	003	Invalid client
F	004	Invalid function
F	998	Duplicate transaction
F	999	System error

8 Appendix A

8.1 DOs and DON'Ts Standard Guidelines in Production

Below are the standard guidelines for all the SP to adhere in the Production:

Types	DO	DON'T
Testing in Production	Send only legitimate transactions to Production environment	Do not perform testing in Production environment
Testing in Production	Dedicate a Test environment to perform testing for further upgrades	Do not perform testing in Production environment
Network	Use stable and enough bandwidth network connection	Do not use unstable connections – message drop causes financial impact for Revenue base transactions
Time Sync	Follow SIRIM time server for all network level to ensure accuracy	Do not use other time server
Release Management	Ensure to check deployment/release version during migration to Production	The Production version does not match with the release version in signed off UAT
Release Management	Ensure all the code changes are locked/freeze after certification of acceptance test	Do not change source code after certification of acceptance test

Types	DO	DON'T
Release Management (Subsequent plazas)	Ensure the Production version is released for subsequent plaza enabled for ABT. SP to ensure the	The Production version from subsequent plaza does not match with the
	quality and correctness of the data	release version in signed off UAT
Operation Management	Ensure to have tools to monitor transaction upload, download & network traffic	No monitoring on network traffic or transactions during day-to-day operation
Security	All the related keys, passwords for API integrations, must not be shared via social messaging/ social media with others (ex: Whatsapp,Skype).	Exposure of the secrets can lead to data tamper that causes financial impact.