2023

Sertag eRetail3.1 System Integration Manual

DALIAN SERTAG TECHNOLOGY CO., LTD

Version History

Version Number	Date	Description	Author	Review
1.0	2022-09-06	Document initialization	Anderson	Anderson
1.4	2023-08-02	Modification of parameters of ESL refresh interface and ESL image refresh interface	Bruce	Bruce
1.5	2023-10-16	eRetail3.1 System Integration Manual Translate	Kris	Tony

Contents

1. Brief description

Applicable to projects based on .NET 6.0 or higher versions.

Applicable to third-generation ESL products.

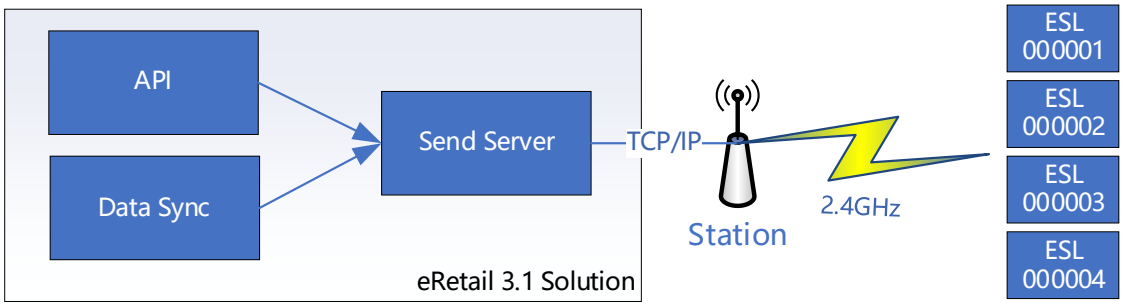
Applicable to second-generation LCD ESL products.

1.1 System Structure

The eRetail3.1 electronic label solution (eRetail3.1) is divided into three components: DataSync / API, Send Server, and hardware devices.

The API is a Restful-style Web API responsible for providing interfaces for customers to proactively push product data. Data Sync is a background service that actively retrieves data from customer systems (such as databases, Web API, Excel). Send Server manages product data, hardware device information, schedules price change tasks, etc. Hardware includes wireless communication base stations, electronic labels, and related network devices.

The overall structure of eRetail3.1 is divided into the following views:



1.2 Comparison Of System Integration Solutions

System integration includes three aspects: data management, template management, and device management. The following table is used to compare the characteristics of integration solutions:

Integration solution	Docking direction	Customer notes	Needn't for attention
Web API: Push product data	Customer System->eRetail	Product data content	Template content Device management
Web API: Push label image	Customer System->eRetail	Product data content Template content	Device management
Data Sync: Provide product data	eRetail->Customer system	Product data content Data source: File/DB/FTP	Device management

1.When customers have development resources but only possess product data, this solution is applicable.

2.When customers have development resources and their systems have label display management capabilities, this solution is applicable;

3.When customers lack development resources but have product data, this solution is applicable.

2. Electronic Shelf Label Interface

When it comes to API addresses in this article, it is assumed that the deployment address of eRetail 3.1 is 192.168.1.92 and the port is 5000.

Note: HTTPS is recommended, HTTP is not recommended.

Please see **Enforce HTTPS in ASP.NET Core | Microsoft Docs**.

2.1 Test Interface

Usage: Check whether the eRetail 3.1 API can be accessed.

HTTP GET

URL: http:// 192.168.1.92:5000/api/hello

Return: "OK"

Note: This interface is used for connectivity testing, informal scenarios.

2.2 Handshake Interface

Purpose: Used for authentication, access to all subsequent interfaces relies on the data obtained from this interface.

HTTP POST

URL: http:// 192.168.1.92:5000/api/login

Content-Type: application/json

Request parameters:

Parameter name	type	describe
userName	String	username
password	String	password

Return format:

Parameter name	Type	Explanation
code	Int	0: Success 4004:Username or password is wrong 6001:Role not exists 4003:User not exists
message	String	success or error message
body	String	Token for subsequent sessions

Note: After obtaining the Token, you need to add this content to the header of subsequent HTTP requests. like:"Authorization: Bearer {token}", valid for six hours.

Example:

Request

```
{
  "userName":"admin",
  "password":"Pass99"
```

```
}
```

Response

```
{
  "code": 0,
  "message":"success",
  "body":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
}
```

2.3 Product Data Interface

Purpose: This interface is used to add and update multiple product data.

HTTP POST

URL: http:// 192.168.1.92:5000/api/goods/saveList?NR=false

Content-Type: application/json

Url Parameters

Parameter name	Type	Describe
NR	String	true / false Whether notification is required to update the ESL, the default is true

Request Parameters:--Array

Parameter name	Type	Describe
shopCode	String	Customer/system store number, you need to create the corresponding store in the system store list first
template	String	Template name, you need to create the corresponding template in the template list first
items	Array String[27]	Product data attributes, temporarily fixed at 27(Scalable)

Note: For store and template creation, Please contact our engineers.

Return format:

Parameter name	Type	Describe
code	Int	0: Success Parameter verification 400: Parameter error 1001: Shop is not exists 9801:is null ShopCode 9801:upc index too long 9801:Cannot find goods code index 9801:Goods name index too long
message	String	success or error message
body	Json Node	message body

Example:

http:// 192.168.1.92:5000/api/goods/saveList?NR=false

Request

```
[
  {
    "shopCode":"0001",//Customer store number
    "template":"REG",// ESL type: REG-Regular price, SAL-Sale price, NOR-Out of Stock, MER-Member..
    "items": [
      "A050",//Customer store number
      "123456",// Product unique code: The product number or barcode can be used
      "Product 1",//Product name
      "123456789012",//UPC1: product barcode 1 or weighing code
      "123456789013",// UPC2: product barcode 2 or weighing code
      "123456789014",// UPC3: product barcode 3 or weighing code
      "8.98",//Price 1: Retail price or Original price
      "8.95",//Price 2: Promotional price
      "8.96",//Price 3: Member price
      "Shanghai",//Origin
      "300ml",//Spec
      "bottle",//Unit
      "qualified",//Grade
      "2021/12/20",//Promotion start date
```

```
        "2021/12/25",//Promotion end date
        "http://www.baidu.com",//QR code
        "Zhang San", //Price employee
        "5.1", //Inventory
        "Extension 1", //Extended field 1 Don't pass empty ""
        "Extension 2", //Extended field 2
        "Extension 3", //Extended field 3
        "Extension 4",
        "Extension 5",
        "Extension 6",
        "Extension 7",
        "Extended 8",
        "Extension 9",
        "Extension 10"
    ]
  }
]
```

http:// 192.168.1.92:5000/api/goods/saveList?NR=false

Request

Response

```
{
  "code": 0,
  "message":"success",
  "body":"121f5151fdfffds21cdf"
}
```

Note: The field properties in this example are fixed by default. In fact, product data field content and templates can be set freely.

2.4 ESL Input Interface

Purpose: This interface is used for batch entry of ESL. This is a system integration-

specific interface used to integrate eRetail 3.1 into customer systems. You also can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/save

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
shopCode	String	System store number
shopCodeCst	String	Customer store number, empty by default. Note: If the system store number and customer store number exist at the same time, the customer store number will be matched first.
tagIdList	StringArray	ESL ID list

Return format:

Parameter Name	Type	Illustrate
code	Int	0: Success Parameter verification 400:Parameter error 3001:Invalid Tag ID
message	String	success or error message
body	String	Default is empty

Example: shopCode or shopCodeCst must upload one of them

Request

```
{
   "shopCode": "0001",
   "tagIdList": [
      "440000010478",
      "440000010479"
   ]
}
```

Response

```
{
   "code": 0,
   "message":"success",
   "body":""
}
```


2.5 ESL Outbound Interface

Purpose: This interface is used for ESL export

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/delete

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
	String Array	ESL ID list

Return format:

Parameter Name	Type	Illustrate
code	Int	0: Success Parameter verification 400:Parameter error 3001:Invalid Tag ID
message	String	success or error message
body	String	Default is empty

Example:

Request


```
["440000010478","440000010479"]
```


Response


```
{
   "code": 0,
```

```
  "message":"success",
  "body":""
}
```

2.6 Label Binding Interface

Purpose: Bind the ESL ID to the product number. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/bind

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
shopCode	String	System store number
ap	String	Can be null - specify base station ID
binds	Array	Binding array
tagID	Array String	ESL ID
goodsCode	Array String	Store number + product number

Return format:

Parameter Name	Type	Illustrate
code	Int	0: Success Parameter verification 400:Parameter error 1001:ESL is null 3001:Invalid Tag ID 3004:Tag is not exists 7001:Goods is not exists
message	String	success or error message
body	String	Default is empty

Example:

Request

```
{
    "shopCode": "0001",
    "ap": "",
    "binds": [{
        "tagID": "40000000EA40",
        "goodsCode": "0001123456"
    }]
}
```

Response

```
{
  "code": 0,
  "message":"success",
  "body":""
}
```

2.7 Label Unbinding Interface

Purpose: Unbind the ESL ID from the product number. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/unbind

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
shopCode	String	System store number
ap	String	Designated base station ID
unbindType	Int	Unbinding type, 0-according to tag ID, 1-according to product number
idList	String Array	Tag ID or product ID, determined according to the unbinding type

Return format:

Parameter Name	Type	Illustrate
code	Int	0: Success Parameter verification 400: Parameter error 1001: ESL or product ID is empty Unbinding ESL 3001: Tag format error 3004: Tag does not exist Unbinding products 7001: Commodity nonexistence 1001: Unbinding failed
message	String	success or error message
body	String	Default is empty

Example:

Request

```
{
   "shopCode":"0001",
   "ap":"",
   "idList":[
      "40000000EA40",
      "40000000EA39"
   ],
  "unbindType": 1//unbind Type 1-ESL ID, 2-product ID
}
```

Response

```
{
   "code": 0,
   "message":"success",
   "body":""
}
```

2.8 Label Combination Binding Interface

Purpose: It is used to combine a label ID with multiple product numbers, Multiple product information can be displayed on a single label. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http://192.168.1.92:5000/api/esl/tag/groupBind

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
shopCode	String	System store number, optional
ap	String	Designated base station ID
template	String	Template type (layout scheme), such as 2X2 2X3wait
tagID	String	Tag ID
goodsCode	Array String	Store number + product number

Return format:

Parameter Name	Type	Illustrate
code	Int	0: Success Parameter verification 400: Parameter error 3001:Invalid Tag ID 3004:Tag is not exists 7001:Goods is not exists 8001:Template not exists 1001:Group Bind Fail
message	String	success or error message
body	String	Default is empty

Example:

Request

```
{
   "shopCode":"0001",
   "ap":"",
   "template":"1x2",
   "tagID":"40000000EA40",
   "goodsCode": [
      "123456",
      "123457"
   ]
}
```


Response


```
{
   "code": 0,
   "message":"success",
   "body":""
}
```


2.9 Label Flash Interface

Purpose: This interface is used to notify the label to flash. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/led

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
shopCode	String	System store number
rgb	string	Flash color (R-Red,G-green,B-Blue)
times	int	Flash time (seconds)
idList	Array	tag or product array
ledType	Int	Flash type (1:Specify tag,0:designated product)

Return format:

Parameter Name	Type	Illustrate
code	Int	0: Success Parameter verification 400: Parameter error 1001:List data is null 1001:LED Fail
message	String	success or error message
body	String	Default is empty

Example:

Request

```
{
    "shopCode":"0001",
    "rgb":"RGB",//R-Red,G-Green,B-Blue
    "times": 10,
    "idList":[
        "40000000EA40" //ledType 1-price tag ID, 0->product ID
    ],
    "ledType": 1
}
```

Response

```
{
    "code": 0,
    "message":"success",
    "body":""
}
```

2.10 Tag Data Push Refresh

Purpose: Used by customers to directly push product data and refresh it to the specified label. The system backend will not guarantee user data.

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/push

Content-Type: application/json

Request parameters:

Parameter Name	Type	Illustrate
shopCode	String	System store number, nullable
shopCodeCst	String	Customer store number, nullable
tagID	string	price tag ID
ap	string	The designated base station sends the default empty
item	Object	Data entity (corresponding to the fields below)
goodsCode	string	Product unique code
goodsName	string	product name
template	string	Template name
items	Character array	Data details array (corresponding fields are used to bind to template display)

Return format:

Parameter Name	Type	Illustrate
code	Int	0: Success Parameter verification 400:Parameter error 3001:Invalid Tag ID 3004:Tag is not exists 2006:product code doesn't exist 8001:Template not exists 1001:Push tag data fail
message	String	success or error message
body	String	Default is empty

Example:

Request

```
{
   "shopCode":"",
   "shopCodeCst":"0002",
   "tagID": "4F000001320A",
```

```
  "ap": "",
  "item": {
    "GoodsCode" : "123456",
    "GoodsName" : "Oberatley Castle dry red",
    "Template" : "SAL",
    "Items" : [
          "0002",
          "9999",
          "Shanghai City",
          "12315",
          "Zhang San",
          "39880",
          "Oberatley Castle dry red",
          "Oberatley Castle dry red",
          "",
          "240209",
          "750ml",
          "bottle",
          "France",
          "828",
          "88888",
          "0",
          "10",
          "qualified"
     ]
}
}
```


Response


```
{
  "code": 0,
  "message":"success",
  "body":""
}
```

2.11 Label Data Batch Push Refresh

Purpose: Used by customers to directly push product data and refresh it to the specified label. The system backend will not guarantee user data.

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/pushList

Content-Type: application/json

Request parameters: push as array

Parameter Name	Type	Illustrate
shopCode	String	System store number, nullable
shopCodeCst	String	Customer store number, nullable
tagID	string	ESL ID
goodsCode	string	Product ID - goodsId is changed to goodsCode
goodsName	string	product name
template	string	Template name
items	StringArray	Data details array (corresponding fields are used to bind to template display)

Return format:

Parameter Name	Type	Illustrate
code	Int	0: success Parameter verification 400:Parameter error 3001:Invalid Tag ID {0} 2006:{0} is not exists 2007:Data {0} is empty 3004:Tag {0} is not exists 8001:Template not exists {0},{1},{2} mistake 1001:Push tag data fail
message	String	success or error message
body	String	Default is empty

Example:

Request

```
[
   {
   "shopCode":"",
   "shopCodeCst":"A050",
   "tagID": "4F000001320A",
   "goodsCode": "123456",
   "goodsName": "product name",
   "template": "Template name",
   "items" : [
            "123456",
            "product name",
            "Shanghai City",
            "12315",
            "Zhang San",
            "39880",
            "Obatley Castle Dry Red",
            "Obatley Castle Dry Red",
            "",
            "240209",
            "750ml",
            "bottle",
            "France",
            "828",
            "88888",
            "0",
            "10",
            "qualified"
     ]
   }
   ....Multiple tags and product data
]
```

Response


```
{
   "code": 0,
   "message":"success",
   "body":""
}
```


2.12 Tag Image Data Push

Purpose: Push the image data directly to the system and send to the tag (note that the image size and the tag should be consistent)

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/Image

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
shopCode	String	System store number, nullable
shopCodeCst	String	Customer store number, nullable
tagID	String	tag ID
base64Data	string	Image base64 string
isDither	Bool	Does the image need to be shake?
batchID	String	nullable, task unique ID
backUrl	String	nullable

Return format:

Parameter Name	Type	Illustrate
code	Int	0: Success Parameter verification 400:Parameter error 3001:Invalid Tag ID {0} 3004:Tag {0} is not exists mistake 1001:Preview Fail 1001:Push Image Error

message	String	success or error message
body	String	Default is empty

Example:

Request

```
{
   "shopCode":"",
   "shopCodeCst":"",
   "tagID":"400001234567",
   "base64Data":"image base64 string",
   "batchID":"",
   "isDither":true
}
```

Response

```
{
   "code": 0,
   "message":"success",
   "body":""
}
```


HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/ImageList

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
	ObjectArray	Push object array
shopCode	String	System store number, nullable
shopCodeCst	String	Customer store number, nullable
tagID	String	tag ID
base64Data	String	Image base64 string
isDither	Bool	Does the image need to be shake?
batchID	String	nullable, task unique ID
backUrl	String	nullable

Return format:

Parameter Name	Type	Describe
code	Int	0: Success Parameter verification 400:Parameter error 3001:Invalid Tag ID {0} 3004:Tag {0} is not exists mistake 1001:Preview Fail 1001:Push Image Error
message	String	success or error message
body	String	Default is empty

Example:

Request

```
[
{
   "shopCode":"",
   "shopCodeCst":"",
   "tagID":"400001234567",
   "base64Data":"image base64 string",
   "batchID":"",
   "isDither":false
},
{
   "shopCode":"",
   "shopCodeCst":"",
   "tagID":"400001234568",
   "base64Data":"image base64 string",
   "batchID":"",
   "isDither":false
}
]
```

Response

```
{
   "code": 0,
```

```
  "message":"success",
  "body":""
}
```


2.13 Tag query

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/query/{id}

Ccontent-Type: application/json

Request parameters:

Parameter Name	Type	Describe
ID	string	tag ID

Return format:

Parameter Name	Type	Describe
code	Int	0: Success
message	String	Success or error message
body	Object	Tag entity
ID	string	Tag ID
apid	string	Last sending base station
shopCode	string	Store number
shopCodeCst	string	Customer store number
tagType	string	Tag type
tagStatus	int	Tag status (0-Not used, 1-normal, 2-Low battery, 3-Communication, 5-Suspected to be lost, 6-ultimately failed)
powerVal	int	Power
temperature	int	Temperature
rssi	int	Signal strength
token	int	Send TOKEN
goodsCode	Array	Bind product ID
totalSend	int	Total sent times
errorCount	int	Total number of sending failures
lastSendTime	string	Last sent time
lastRecvTime	string	Last return time

Example:

Request

```
http://192.168.1.92:5000/api/esl/Tag/query/3600000148E3
```

Response


```
{
   "code": 0,
   "message": "success",
   "body": {
      "apid": "16",
      "shopCode": "0002",
      "tagType": "36",
      "tagStatus": 1,
      "powerVal": 30,
      "temperature": 24,
      "rssi": -89,
      "token": 215,
      "goodsCode": ["10824"],
      "totalSend": 322,
      "errorCount": 106,
      "lastSendTime": "2022-03-10 09:02:34",
      "lastRecvTime": "2022-03-10 09:02:56",
      "id": "3600000148E3",
   }
}
```


2.13 Commodity Inquiry

Purpose: Used to query product information based on customer store number

HTTP POST

URL: http:// 192.168.1.92:5000/api/Goods/getList

Content-Type: application/json

Request parameters:

parameter name	type	Describe
pageIndex	Int	page number
pageSize	Int	Number of pages
sort	String	Nullable
desc	String	asc/desc Nullable
shopCodeCst	String	Customer store number, Nullable
shopCode	String	System store number, Nullable
goodsCode	String	Product unique code, Nullable

Return format:

parameter name	type	Describe
code	Int	0: Success
message	String	success or error message
body	Object	Return entity
totalCount	int	total
itemList	Object Array	Product entity list
id	String	Product primary key
shopCodeCst	String	Customer store number
shopCode	String	System store number
goodsCode	String	Product unique code

goodsName	String	Product name
template	String	Template name
upc	String array	Search code
items	String Array	Product details array, subject to actual data

Example:

Request


```
{
   "pageIndex": 1,
   "pageSize": 100,
   "sort": "",
   "desc": "",
   "shopCodeCst": "012412",
   "shopCode": "",
   "goodsCode": ""
}
```


Response


```
{
   "code": 0,
   "message": "success",
   "body": {
"totalCount": "999",
"itemList": [
{
```

```
"shopCodeCst": "012412",
"id": "00011000786815",
"shopCode": "0001",
"goodsCode": "1000786815",
"goodsName": "A",
"template": "REG",
"upc": [
    "1000786815"
],
"items": [
    "14297",
    "1000786815",
    "A",
    "",
    "6903148083109",
    "0.00",
    "34.90",
    "0",
    "0.00",
    "2.8KG",
    "",
    "",
    "qualified",
    "",
    "",
```

```
            "bag",

            "bag",

            "0518",

            "60081373",

            "",

            "Guangzhou City"

]},

.....

                ]

}

}
```


2.14 Base Station Heartbeat Callback Interface

Purpose: The base station heartbeat is actively pushed to the client system.

Note: This interface needs to be provided by the customer. After feedback is enabled in the system configuration, the base station will call back the interface after heartbeat.

Please use the following format for request parameters

HTTP POST

URL: http://Client interface address/apheart

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
ShopCode	String	System store number
MAC	String	Base station MAC
APID	String	Base station ID
ApStatus	Int	State 0 initialization 1online 3Offline 4at work
LastOnlineTime	Date Time	Last online time

Return format:

Parameter name	Type	Describe
code	Int	0: Success
message	String	success or error message
body	Object	Return entity

Request


```
{
   "ShopCode":"0001",
   "MAC":"111111111",
   "APID":"01",
   "ApStatus":1
   "LastOnlineTime":"2020-05-01 11:11:59"
}
```


Response


```
{
   "code": 0,
   "message":"success",
   "body":""
}
```


2.15 Tag Feedback Callback Interface

Purpose: Tag is refreshed successfully or failed, it will be actively pushed to the customer system.

Note: This interface needs to be provided by the customer. After system configuration enables feedback, the system will call back this interface after receiving tag feedback. Please use the following format for request parameters.

HTTP POST

URL: http://Client interface address/callback

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
ShopCode	String	System store number
TagID	String	Tag ID
TagType	String	Tag type
BatchID	String	Send batch number
APID	String	Sending base station ID
GoodsCode	String	Commodity ID
GoodsName	String	Product name
TempType	String	Template name
TaskStatus	Int	Task status 0 initialization 1 Communication 2 success 4 fail 5 time out 8 cover
TaskType	Int	Task type
Rssi	Int	Signal strength
PowerVal	Double	Power
Temperature	Double	Temperature
LastRecvTime	Date Time	Last feedback time

Return format:

Parameter Name	Type	Describe
code	Int	0: success, other: error
message	String	success or error message
body	Object	Return entity

Request

```
{
   "ShopCode": "0001",
   "TagID": "401234567890",
   "TagType": "40",
   "BatchID": "1323dwafs11343",
   "APID": "01",
   "GoodsCode": "123456",
   "GoodsName": "test",
   "TempType": "REG",
   "TaskStatus": 1,
```

```
    "TaskType": 1,
    "Rssi": -56,
    "PowerVal": 32,
    "Temperature": 32,
    "LastRecvTime": "2020-05-01 11:11:59"
}
```


Response


```
{
    "code": 0,
    "message": "success",
    "body": ""
}
```


2.16 Tag Refresh (Specified Tag or Whole Store)

Purpose: Used to refresh the screen with tags and will not change the display content.

HTTP POST

URL: http:// 192.168.1.92:5000/api/esl/tag/Refresh

Content-Type: application/json

Request parameters:

Parameter Name	Type	Required	Describe
shopCode	String	yes	System store number
refreshType	Int	yes	Refresh type 1: template, 2: tag type, 3: specified store, 4: tag ID list
refreshName	String	no	When refreshType=1, it is the template name, when refreshType=2, it is the tag type, and the others are empty.
tags	String Array	no	When refreshType=4, specify the array of tag IDs. Others are empty

Return format:

Parameter Name	Type	Describe
code	Int	0: Success

		Others are failures
message	String	success or error message
body	String	Default is empty

Example:

Request

```
{
   "shopCode": "0001",
   "refreshType": 4,
   "refreshName": "",
   "tags": [
"401234567890",
"401234567891"
   ]
}
```

Response

```
{
   "code": 0,
   "message":"success",
   "body":""
}
```


3. Store Management

3.1 Store Creation

Purpose: Create a store

HTTP POST

URL: http:// 192.168.1.92:5000/api/shop/add

Content-Type: application/json

Request parameters:

Parameter Name	Type	Required	Describe

shopCode	String	yes	System store number
shopCodeCst	Int	yes	Customer store number
shopName	String	yes	Store name
shopAddress	String Array	no	Store address

Return format:

Parameter Name	Type	Describe
code	Int	0: Success Others are failures
message	String	success or error message
body	String	Default is empty

Example:

Request


```
{
    "shopCode": "0001",
    "shopCodeCst": "A001",
    "shopName": "Test store",
    "shopAddress": "Suzhou"
}
```

Response

```
{
    "code": 0,
    "message":"success",
    "body":""
}
```

3.2 Store Deletion

Purpose: Delete a store. Deleting a store will cause the original data to be lost. Please use it with caution.

HTTP GET

URL: http:// 192.168.1.92:5000/api/shop/delete/{id}

Content-Type: application/json

Request parameters:

Parameter name	Type	Required	Describe
ID	int	yes	Refer to 3.3 for store primary key ID, which can be obtained through store query.

Return format:

Parameter Name	Type	Describe
code	Int	0: Success Others are failures
message	String	Success or error message
body	String	Default is empty

Example: http:// 192.168.1.92:5000/api/shop/delete/1

Request

Response

```
{
   "code": 0,
   "message":"success",
   "body":""
}
```

3.3 Store Inquiry

Purpose: Query the relationship between all stores under the current user

HTTP GET

URL: http:// 192.168.1.92:5000/api/shop/queryShopListByUser

Content-Type: application/json

Request parameters:

Parameter name	Type	Describe

Return format:

Parameter Name	Type	Describe
code	Int	0: Success
message	String	success or error message
body	Object Array	Store list entity
shopCode	String	System store number
shopCodeCst	String	Customer store number
shopName	String	Store name
shopAddress	String	address
shopStatus	String	Status: 0 normal, 1 closed
ID	Int	Store primary key ID

Note: After obtaining the Token, you need to add this content to the header of subsequent HTTP requests. like:"Authorization: Bearer {token}", valid for six hours.

Example:

Request

Response


```
{
   "code": 0,
   "message": "success",
   "body": [
      {
         "shopCode": "0001",
         "shopCodeCst": "A050",
         "shopName": "23 test store",
         "shopAddress": "sz",
         "shopStatus": 0,
         "id": 22
```

```
    },
    {
      "shopCode": "0002",
      "shopCodeCst": "0002",
      "shopName": "0002",
      "shopAddress": "0002",
      "shopStatus": 0,
      "id": 29
    }
  ]
}
```


4. Digital Signage Interface

4.1 Signage List Query Interface

Purpose: Used for sign list query. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. If not necessary, you can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/tft/tft/queryList

Content-Type: application/json

Request parameters:

Parameter name	Type	Describe
TFTId	String	Device Id
shopCode	String	Store number
shopCodeCst	String	Customer store number
pageIndex	Int	Page index
pageSize	Int	Page size

Return format:

Parameter name	Type	Describe

code	Int	0: Success Parameter verification 400:Parameter error
message	String	success or error message
Body	String	Default is empty
totalCount	Int	Query the number of items
items	Array	Query result list
tftId	String	Device Id
custId	String	Customer Id
instId	String	Entity Id
tftType	String	Equipment type
tftVersion	String	Device version number
lastHeartbeatTime	DateTime	Last heartbeat time
lastOfflineTime	DateTime	Last offline time
lastOnlineTime	DateTime	Last online time
lastUpdatedTime	DateTime	Last updated time
createdTime	DateTime	Creation time
tftToken	String	Communication token
videoTaskId	String	Task Id
shopCode	String	Store number
shopCodeCst	String	Customer store number
goods	Array	Product ID (array)
templateName	String	Template name
tftBindRes	String	Device binding resources
areaId	Int	AreaId
ssid	String	WiFi SSID
ssidpwd	String	Wifi SSID Password
serverUrl	String	Service address
certUrl	String	Certificate address
certMD5	String	Certificate MD5 verification
status	Int	Device status identification: 0 Offline 1 Online 2 Downloading 3 Inactive
tftName	String	Device name
firmware	String	Firmware version
lastVideoTaskId	String	Last task ID
firstDownloadTime	DateTime	First download time
lastDownloadTime	DateTime	First download time
lastConfirmTime	DateTime	Last confirmed time
downLoadCount	Int	download times
confirmCount	Int	Number of confirmations
heartBeatCount	Int	Heartbeat statistics
taskCount	Int	total number of tasks
lastSpanTime	DateTime	The last task took (seconds)

areaName	String	area name
statusStr	String	Device status name
goodsStr	String	Product name (array)
lastHeartbeatTimeStr	String	Last heartbeat time (MM-dd HH:mm)
lastDownloadTimeStr	String	Last download time (MM-dd HH:mm)
totalCount	Int	Query the number of items

Example:

Request


```
{
   "TFTId": "",
   "shopCode": "0001",
   "pageIndex": 1,
   "pageSize": 30
}
```


Response


```
{
      "code": 0,
      "message": "success",
      "body": {
            "totalCount": 149,
            "items": [
                  {
                        "tftId": "C49C14844658",
                        "custId": "ETAG",
                        "instId": "0001",
                        "tftType": "5C",
                        "tftVersion": "61",
                        "lastHeartbeatTime": "2022-09-06 13:17:09",
                        "lastOfflineTime": "1900-01-01 00:00:00",
                        "lastOnlineTime": "1900-01-01 00:00:00",
                        "lastUpdatedTime": "2022-09-06 13:17:09",
                        "createdTime": "2022-09-02 19:24:25",
                        "tftToken": "",
                        "videoTaskId": "b88d7987-4ad3-462a-9aa7-
```

```
f917f7267d82",
                    "shopCode": "0001",
                    "goods": [
                        "1"
                    ],
                    "templateName": "Display area video",
                    "tftBindRes": [],
                    "areaId": 7,
                    "ssid": "",
                    "ssidpwd": "",
                    "serverUrl": "",
                    "certUrl": "",
                    "certMD5": "",
                    "status": 1,
                    "tftName": "",
                    "firmware": "",
                    "lastVideoTaskId": "b88d7987-4ad3-462a-9aa7-
f917f7267d82",
                    "firstDownloadTime": "2022-09-06 13:16:10",
                    "lastDownloadTime": "2022-09-06 13:16:10",
                    "lastConfirmTime": "2022-09-06 13:17:02",
                    "downLoadCount": 152,
                    "confirmCount": 152,
                    "heartBeatCount": 257,
                    "taskCount": 152,
                    "lastSpanTime": 52,
                    "areaName": "video",
                    "statusStr": "",
                    "goodsStr": "",
                    "lastHeartbeatTimeStr": "",
                    "lastDownloadTimeStr": ""
                }
            ]
        }
}
```

4.2 Signage Query Interface

Purpose: Used for signage query. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. If not necessary, you can directly use the backend management system or App of eRetail 3.1.

HTTP GET

URL: http:// 192.168.1.92:5000/api/tft/tft?tftId=C49C14844658

Content-Type: application/json

Request parameters:

Parameter name	Type	Describe
TFTId	String	Device Id

Return format:

Parameter name	Type	Describe
code	Int	0: Success 400: Parameter error
message	String	success or error message
body	String	Default is empty

Example:

Request

```
TFTId =C49C14844658
```

Response

```
{
    "code": 0,
    "message": "success",
    "body": {
        "tftId": "C49C14844658",
        "custId": " ETAG",
        "instId": "0001",
```

"tftType": "5C",
"tftVersion": "61",
"lastHeartbeatTime": "2022-09-06 13:17:09",
"lastOfflineTime": "1900-01-01 00:00:00",
"lastOnlineTime": "1900-01-01 00:00:00",
"lastUpdatedTime": "2022-09-06 13:17:09",
"createdTime": "2022-09-02 19:24:25",
"tftToken": "",
"videoTaskId": "b88d7987-4ad3-462a-9aa7-f917f7267d82",
"shopCode": "0001",
"shopCodeCst": "A050",
"goods": [
"00011"
],
"templateName": "Display area video",
"tftBindRes": [],
"areaId": 7,
"ssid": "",
"ssidpwd": "",
"serverUrl": "",
"certUrl": "",
"certMD5": "",
"status": 1,
"tftName": "",
"firmware": "",
"lastVideoTaskId": "b88d7987-4ad3-462a-9aa7-f917f7267d82",
"firstDownloadTime": "2022-09-06 13:16:10",
"lastDownloadTime": "2022-09-06 13:16:10",
"lastConfirmTime": "2022-09-06 13:17:02",
"downLoadCount": 152,
"confirmCount": 152,
"heartBeatCount": 257,
"taskCount": 152,
"lastSpanTime": 52,
"areaName": "video",

```
            "statusStr": "",
            "goodsStr": "",
            "lastHeartbeatTimeStr": "",
            "lastDownloadTimeStr": ""
        }
}
```


4.3 Signage Preview Interface

Purpose: Used for signage preview. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems.You can directly use the backend management system or App of eRetail 3.1.

HTTP GET

URL: http:// 192.168.1.92:5000/api/tft/tft/PreviewTft/C49C14844658

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
TFTId	String	Device Id

Return format:

Parameter Name	Type	Describe
code	Int	0: Success 404: Resource does not exist 1001: The device has not been bound to a template
message	String	Success or error message
body	String	Default is empty

Example:

Request

```
TFTId =C49C14844658
```

Response


```
{
    "code": 0,
    "message": "success",
    "body": [
        {
            "id": 0,
            "createdBy": null,
            "createdTime": "0001-01-01 00:00:00",
            "mac": null,
            "displayId": 0,
        }
]
}
```


4.4 Sign Template Query Name Interface

Purpose: This interface is used to query the name of the sign template. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. If not necessary, you can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/template/queryTemplateByName

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
groupTemp		Whether to combine templates 0 all, 1 alone, 2 combined
shopCode	String	Store number
tagType	String	Device model
templateName	String	Template name

Return format:

Parameter Name	Type	Describe
code	Int	0: Success 400: Parameter error
message	String	success or error message
body	String	Default is empty

Example:

Request


```
{
    "groupTemp": 0,
    "shopCode": "0001",
    "tagType": "5C",
    "templateName": "Display area video"
}
```


Response


```
{
    "code": 0,
    "message": "success",
    "body": [
        {
            "templateName": "Display area video",
            "tagType": "5C",
            "shopCode": "0001",
            "templateType": 1,
            "templateStatus": 0,
            "blocks": 2,
            "preview":
            "api/template/preview_template?shopCode=0001&tagType=5C&name=%e6%98%be%e7%a4%ba%e5%8c%ba%e5%9f%9f%e8%a7%86%e9% a2%91&timeStamp=1661614464253",
            "templateData": null,
            "id": 60,
            "isDeleted": false,
            "createdBy": "System",
```

```
                "createdTime": "2022-08-16 16:58:12",
                "lastUpdatedBy": "System",
                "lastUpdatedTime": "2022-08-27 23:34:24"
            }
        ]
}
```


4.5 Sign Template Query ID Interface

Purpose: Used to query the ID of the sign template. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.

HTTP GET

URL: http:// 192.168.1.92:5000/api/template/query/60

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
ID	Int	Template Id

Return format:

Parameter Name	Type	Describe
code	Int	0: Success
message	String	success or error message
body	String	Default is empty

Example:

Request

Response

```
{
    "code": 0,
    "message": "success",
    "body": {
        "templateName": "Display area video",
        "tagType": "5C",
        "shopCode": "0001",
        "templateType": 1,
        "templateStatus": 0,
        "blocks": 2,
        "preview":
        "api/template/preview_template?shopCode=0001&tagType=5C&name=%
        e6%98%be%e7%a4%ba%e5%8c%ba%e5%9f%9f%e8%a7%86%e9%
        a2%91&timeStamp=1661614464253",
        }
}
```


4.6 Signage Binding Interface

Purpose: Used for signage binding. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.\

HTTP POST

URL: http:// 192.168.1.92:5000/api/tft/tft/bind

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
areaId	Int	Area Id
bindRes	Entity	A list of position entities for the template element
displayIndex	Int	screen index (default 0)
goods	Array	Product array
shopCode	String	Store number; nullable. Note: Store number and customer store number are required.
templateName	String	Template name

tftId		Device Id

Return format:

Parameter Name	Type	Describe
code	Int	0: Success Parameter verification 400:Parameter error 1100:tag not fond mistake 3201:goods {item} is not exist 1001:Binding error
message	String	success or error message
body	String	Default is empty

Example:

Request


```
{
        "areaId": 7,
        "bindRes": [],
        "displayIndex": 0,
        "goods": [
        "00011"
        ],
"shopCode": "0001",
"shopCodeCst": "A050",
        "templateName": "Display area video",
        "tftId": "C49C14844658"
}
```


Response


```
{
        "code": 0,
        "message": "success",
        "body": ""
}
```

4.7 Signage Unbinding Interface

Purpose: Used to unbind the sign. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/tft/tft/unbind

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
idList	Array	Device array
shopCode	String	Store number; nullable. Note: Store number and customer store number are required.

Return format:

Parameter Name	Type	Describe
code	Int	0: Success Parameter verification 400:Parameter error 3102:tft is null
message	String	Success or error message
body	String	Default is empty

Example:

Request

```
{
    "idList": [
    "F49520998244"
    ],
"shopCode": "0001"
}
```

Response

```
{
    "code": 0,
    "message": "success",
    "body": ""
}
```

4.8 Signage Screen/Bright Screen Interface

Purpose: Used for signage display/light screen. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/tft/task/switchTask

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
tftIds	Array	Device array
taskCode	Int	Task code 11: Turn off the backlight, 12:Light up the backlight

Return format:

Parameter Name	Type	Describe
code	Int	0: Success Parameter verification 400:Parameter error mistake 1001:TFTIds is not null 1001: Switch Task Settings HOST is null 1001: Swtich Task TFTIds is error
message	String	Success or error message
body	String	Default is empty

Example:

Request

```
{
    "tftIds": [
        "C49C14844658"
    ],
    "taskCode": "11"
}
```

Response

```
{
    "code": 0,
    "message": "success",
    "body": ""
}
```

4.9 Signage Refresh Interface

Purpose: Used for signage refresh. This is a system integration-specific interface used to integrate eRetail 3.1 into customer systems. You can directly use the backend management system or App of eRetail 3.1.

HTTP POST

URL: http:// 192.168.1.92:5000/api/tft/tft/refresh

Content-Type: application/json

Request parameters:

Parameter Name	Type	Describe
tftId	String	Device Id
shopCode	String	Store number

Return format:

Parameter Name	Type	Describe
code	Int	0: Success Parameter verification 400:Parameter error mistake 1001: No items selected

		1001: refresh error
message	String	Success or error message
body	String	Default is empty

Example:

Request

```
{
    "tftId": [
    "F4952099968E"
    ],
    "shopCode": "0001"
}
```

Response

```
{
    "code": 0,
    "message": "success",
    "body": ""
}
```


5. Data Synchronization

Data synchronization methods: based on database/intermediate table/view query statements or stored procedures; based on format files (such as Excel, CSV/TXT, XML, JSON, etc.); based on FTP file systems.

5.1 D2M Dynamic Data Model

For simple data structures, a dynamic model (D2M) can be used for data synchronization. 1) Data structure definition 2) Template switching logic definition, see eRetail D2M configuration page.

5.2 Customized Development

For cases where custom development is required and the above integration solutions cannot effectively address the customer's system integration needs, please contact our project department personnel for specific solutions and further information.