API Guide

Get Started
To connect and upload your Point-of-Sales daily sales data to the server. You need to use our **Web API** provided with http **POST method** to our server URL.

Step 1:
There is only one API called needed for uploading all the sales info to server. You will need to compute all the required info into a single **JSON** data string.

To understand how real figures are computed into the JSON, you may refer to the example Sales Figure below first.

Store Info

Parameter	Value	Description
contract_no	CSLEASE-006577	This code can be obtained at My Store page
sales_date	2020-04-06	This is the date of the Sales you wish to upload. It must be in YYYY-MM-DD format.
store_sales_amount	822.17	Entire Store Sales amount with sum up from all Cashier Counters' "cashier_amount"'. You can provide up to 2 decimal point and **DO NOT** include with thousand seperator(,)

Cashier #1

Parameter	Value	Description
cashier_counter	C1	You can compute any String value without any duplication with the other Cashier Counter Code
discount_amount	15	Sales amount from this Cashier Counter. You can provide up to 2 decimal point and **DO NOT** include with thousand seperator(,)
before_tax_amount	219.13	Amount after discount before apply any Tax. You can provide up to 2 decimal point and **DO NOT**

		include with thousand seperator(,)
tax_amount	13.15	Tax Amount (eg. SST 6%). You can provide up to 2 decimal point and **DO NOT** include with thousand seperator(,)
cashier_amount	232.28	Total Amount collected from this Cashier. You can provide up to 2 decimal point and **DO NOT** include with thousand seperator(,)
Purchase Type	**cashier_amount = product + dine_in + take_away**	
product	106.12	Total Amount of Item(Product/Services) purchased. You can provide up to 2 decimal point and **DO NOT** include with thousand seperator(,)
dine_in	116.02	Total Amount by Dine-In mode. You can provide up to 2 decimal point and **DO NOT** include with thousand seperator(,)
take_away	16.14	Total Amount by Take Away mode. You can provide up to 2 decimal point and **DO NOT** include with thousand seperator(,)
Payment Type		
payment_mode	CASH: 112.46 BOOST: 119.82 ...	Amount by Payment mode. You can provide as many entries as you have. Use CASH for cash mode. You can provide up to 2 decimal point and **DO NOT**

		include with thousand seperator(,) You can use any alphanumeric text (small/capital) as `mode` for the rest of the payment modes you have.

Cashier #2, #3, #4...

You can repeat the same structure for 2nd, 3rd cashier counters onwards.

Upon understanding what are the figures and reporting structure required, you may start to compute your daily Sales figure into a JSON formatted string. A beautify version is provided below for your reference.

Sample JSON Data


```
{"contract_no":"CSLEASE-006577","sales_date":"2020-04-
06","store_sales_amount":"822.17","cashier":[{"cashier_counter":"C1","discount_am
ount":"15","before_tax_amount":"219.13","tax_amount":"13.15","cashier_amount":"23
2.28","product":"100.12",
"dine_in":"116.02","take_away":"16.14","payment_mode":[{"mode":"CASH","amount":"1
12.46"},{"mode":"BOOST","amount":"119.82"}]},{"cashier_counter":"C2","discount_am
ount":"15","before_tax_amount":"556.5","tax_amount":"33.39","cashier_amount":"589
.89","product":"100.60","dine_in":"17.38","take_away":"471.91","payment_mode":[{"
mode":"CASH","amount":"406.83"},{"mode":"BOOST","amount":"183.06"}]}]}
```


Sample JSON data above is computed based on a single day sales for 2 cashier counters, which accept Cash and Boost two payment mode.

The JSON data also consist of total discount amount given, before tax amount & tax amount within a single day sales.

Step 2:

Once you have completed your JSON computation, the next step is understanding the **HASH** method use in the API call.

To compute the **HASH** value, we use a widely available MD5 encryption onto your JSON and SECRET. You may refer to the formula below for the HASH computation.

First, you need to **concatenate** JSON and SECRET and follow by a **MD5 encryption** to obtain the HASH string value.

Noted: You must ensure your json string is minified (no blank space).

Hashing Formula

```
HASH = md5( JSON + SECRET )
```

Sample PHP Code

```
$hash = md5($json.$secret);
```

You may obtained your SECRET from the Credential Page.

Conclusion:
At this point of time, you should have understand your JSON, SECRET & HASH value. You will need to use the JSON & HASH value when connecting to our API server. Before hand, you also need to obtain api KEY from the credential as well.

You should have these now:

- JSON
- HASH
- KEY

Connect to API Server

You need to connect to API server to submit the 3 values you have computed (JSON, HASH & KEY) now.

Prepare Parameters

API Call URL

https://api.citysqpos.com/sales/

To execute the API call, you need to perform a web `https` request with `POST` method to the above URL.

Parameter in your https POST

Post Field	Value	Description

json	{...}	The JSON data string you had learnt to compute in Get Started guide. Noted: You must ensure your json string is minified (no blank space).
key	5f11984d24e4d4.80690057	The KEY value that provided in Credential page
hash	md5(JSON+SECRET)	The HASH data string you had learnt to compute in Get Started guide. You can get your SECRET at Credential page.

Response that You will Get

You will be replied with `json result` after you performed the API Call with success `status 200` web connection.

Response Sample when Success upload a day sales

```
{    "status": "SUCCESS",    "message": "Sales has been uploaded successfully",
"key_mode": "PRODUCTION",    "error_count": "0",    "error_message": "",
"json": {...}}
```

About the JSON Reply

Key	Possible Value(s)	Description
status	SUCCESS, FAIL	These 2 possible value is the first thing you interpreted to determined if your call is successfully performed.
message	*string*	This is high level description of your API call status.
key_mode	PRODUCTION, DEVELOPMENT	Base on which KEY + SECRET combination you used during your API call and hashing.
error_count	*integer*	If the API call is FAIL, this value show you total of error(s) occurred.
error_message	*Array*	If the API call is FAIL, this value show you list of error message(s) in array mode.

json	*{...} json string*	This will always reply you exactly the json string you submitted.