Push API: Permit Create

The **Push API: Permit Create** allows OPSCOM administrators to integrate external parking systems directly with the OPSCOM Controller. This RESTful integration automatically synchronizes paid or temporary permit details from third-party hardware and software, such as parking meters, pay stations, and mobile parking applications. By pushing active parking instances into the centralized database, administrators ensure real-time compliance tracking and seamless enforcement across all connected devices.

Setup and Configuration

Before third-party systems can transmit transaction data into the system, specific environmental settings and authorization keys must be established.

> API access is a premium paid feature that requires licensing activation. Organizations must contact their OPSCOM Account Executive to negotiate access and enable the endpoint interface for their environment.

Admin Side Configuration

1. Secure an authenticated API token from your OPSCOM technical representative.
2. Confirm that all target parking zones match your external hardware naming conventions. Review configuration rules on the *Parking Zone Administration* page to avoid disconnected reporting errors.
3. Distribute the production endpoint URL and unique alphanumeric token to your third-party integration developers or vendors (e.g., HotSpot).

Using this Feature

The integration relies on an external system triggering HTTP POST requests to the centralized server. The endpoint accepts a standard structured JSON payload containing authorization, vehicle identity, and time boundaries.

Endpoint Address

`POST [https://controller.operationscommander.io/api/OC-TOMA/v1/permits/push](https://controller.operationscommander.io/api/OC-TOMA/v1/permits/push)`

Request Header Requirements

External systems must supply the following exact headers within every network transmission:

- `Content-Type: application/json`
- `Accept: application/json`

Request Payload Examples

Raw HTTP Request

HTTP

```
POST /api/OC-TOMA/v1/permits/push HTTP/1.1
Host: controller.operationscommander.io
Accept: application/json
Content-Type: application/json
Cache-Control: no-cache

{
    "apiToken": "YOUR-API-TOKEN",
    "Amount": "14.50",
    "CurrencyID": "CAD",
    "LicencePlate": "PL8RDR",
```

```
    "zone": "Lot 4",
    "permitNo": "L4-1138",
    "startTime": "2018-07-02T09:00:00",
    "endTime": "2018-07-02T09:30:00",
    "promoCode": "DISCOUNT20"
}
```

JavaScript XMLHttpRequest Example

JavaScript

```
var request = new XMLHttpRequest();

request.open('POST', 'https://controller.operationscommander.io/api/OC-TOMA/v1/permits/push');

request.setRequestHeader('Content-Type', 'application/json');
request.setRequestHeader('Accept', 'application/json');

request.onreadystatechange = function () {
  if (this.readyState === 4) {
    console.log('Status:', this.status);
    console.log('Headers:', this.getAllResponseHeaders());
    console.log('Body:', this.responseText);
  }
};

var body = {
    "apiToken": "YOUR-API-TOKEN",
    "Amount": "14.50",
    "CurrencyID": "CAD",
    "LicencePlate": "PL8RDR",
    "zone": "Lot 4",
    "permitNo": "L4-1138",
    "startTime": "2018-07-02T09:00:00",
    "endTime": "2018-07-02T09:30:00",
    "promoCode": "DISCOUNT20"
};
```

```
request.send(JSON.stringify(body));
```

Request Object Attributes

The endpoint processes incoming data objects using strict key names. Use the definitions below to map external database variables correctly.

Attribute	Type	Limits	Possible Names	Required / Optional	Description
apiToken	String	50-character alphanumeric including dashes	apiToken	**Required**	The unique system token supplied by OPSCOM for validation.
LicensePlate	String	25-characters	plate LicencePlate	**Required**	The license plate of the vehicle receiving the parking permission.

Attribute	Type	Limits	Possible Names	Required/ Optional	Description
Amount	String	9-character decimal	amount Amount	**Optional**	Transaction amount. Must contain at least 3 digits, two of which are penny values. The minimum allowable value is $0.01, and the maximum allowable value is $999999.99.
CurrencyID	String	10-characters	currency CurrencyID	**Optional**	Transaction currency type. Supported defaults include CAD or USD.

Attribute	Type	Limits	Possible Names	Required/ Optional	Description
Start Date	String	20-characters	startTime StartDateUtc	**Required**	Must be in the format of Y-m-d\TH:i:s e.g. 2000-05-30T14:38:22 For formatting help, see PHP Date Formatting
End Date	String	20-characters	endTime EndDateUtc	**Required**	Must be in the format of Y-m-d\TH:i:s e.g. 2000-05-30T14:38:22 For formatting help, see PHP Date Formatting
permitNo	String	50-characters	permitNo	**Optional**	The permit identifier or unique receipt ID

Attribute	Type	Limits	Possible Names	Required/ Optional	Description
Ticket Number	String	50-characters	TicketNumber	**Optional**	The ticket identifier or unique receipt ID
zone	String	200-characters	zone ParkingZoneName	**Optional**	The localized parking area identifier. This should match an explicit string name inside the system. If the zone does not match a zone in our system, it will be a disconnected record and may not report properly.
promoCode	String	50-characters	promoCode	**Optional**	Alphanumeric validation string detailing applied customer discounts, validation codes, or free parking promotional campaigns (e.g., HotSpot promo keys).

API Server Responses

Successful Response

Upon successfully accepting and writing a record to the database, the server transmits an HTTP status code `200 OK` along with a tracking object.

Successful Response

The response will be a json object. Content-Type: application/json

JSON


```
{
    "status": "success",
    "reference_id": "1a9b5375-cb75-4c71-9939-eeae550b09ac",
    "InternalReferenceID": "1a9b5375-cb75-4c71-9939-eeae550b09ac"
}
```


Best Practices and Considerations

- **Audit third-party zone names precisely.** Ensure that zone names passed into the **zone** parameter exactly match your current system setup. If an external pay station pushes a value that does not correspond with a recognized lot name, it registers as a disconnected record and fails to populate correctly within enforcement reports.
- **Validate promotional campaign metrics.** When working with mobile applications like HotSpot, instruct vendors to map their promotional parameters to the **promoCode** field. Administrators can track usage and verify free or discounted transaction validity by auditing the dedicated promotional code column found on the *Paystation Status* page.

- **Adhere strictly to UTC string constraints.** Confirm that your developers match the ISO-8601 date formatting style explicitly for **startTime** and **endTime**. For technical validation, reference the *PHP Date Formatting* development rules.

Revision #4
Created 9 October 2024 08:24:24
Updated 2 September 2026 12:37:42