Onebee Access Control System

API Documentation

Version 1.0

Generated on: April 08, 2025

Onebee Access Control System API Documentation

Table of Contents

Onebee Access Control System API Documentation

1. Introduction

The Onebee Access Control System API provides a comprehensive set of endpoints for managing access control operations. This API allows you to manage users, locations, lanes, devices, and access logs programmatically.

2. Interactive API Documentation (Swagger UI)

The Onebee Access Control System provides interactive API documentation through Swagger UI. This interactive documentation allows you to explore and test the API directly from your browser.

Accessing Swagger UI

The Swagger UI documentation is available at:

Production: https://api.onebee.com/docs/

Development: http://localhost:5000/docs/

Features of Swagger UI

- Interactive API Testing: Try out API endpoints directly from the browser
- Request/Response Examples: See example requests and responses for each endpoint
- Authentication Integration: Test authenticated endpoints with your JWT token
- Schema Definitions: View detailed models and data structures
- OpenAPI Specification: Download the complete API specification in OpenAPI format
- Real-time Validation: Validate your requests before sending them

API Namespaces

- vehicle: Vehicle detection and ANPR operations
- barrier: Barrier control operations
- health: Hardware health check operations

10. Using Swagger UI

Authentication in Swagger UI

1. Click the "Authorize" button at the top of the page
2. Enter your JWT token in the format: Bearer <your_token>
3. Click "Authorize"
4. All subsequent requests will include your authentication token

Testing Endpoints

1. Expand the endpoint you want to test by clicking on it
2. Click the "Try it out" button
3. Fill in the required parameters
4. Click "Execute"
5. View the response below the request

Available Models

- VehiclePresence: Vehicle detection data
- ANPRResult: ANPR recognition results
- BarrierControl: Barrier control commands
- User: User management data
- Location: Location management data
- Lane: Lane configuration data
- Device: Device management data

- AccessLog: Access log entries

3. Authentication

All API requests require authentication using a JWT token. Include the token in the Authorization header of your requests:

Authorization: Bearer <your_token>

4. Base URL

Production: https://api.onebee.com/v1

Development: http://localhost:5000/api

5. API Endpoints

5.1 User Management

GET /users

Retrieves a list of all vehicle users.

Example Request:

GET /api/users

Authorization: Bearer <your_token>

Example Response:

```
{
    "users": [
        {
            "id": 1,
            "name": "John Doe",
            "vehicle_number": "KA01AB1234",
            "fastag_id": "FASTAG1234",
            "location_id": 1,
            "valid_from": "2024-01-01",
            "valid_to": "2024-12-31",
            "is_active": true
        }
    ]
}
```

Onebee Access Control System API Documentation

POST /users

Creates a new vehicle user.

Example Request:

POST /api/users

Authorization: Bearer <your_token>

Content-Type: application/json


```
{
    "name": "Jane Doe",
    "vehicle_number": "MH02CD5678",
    "fastag_id": "FASTAG5678",
    "location_id": 2,
    "valid_from": "2024-01-01",
    "valid_to": "2024-12-31",
    "is_active": true
}
```

5.2 Location Management

GET /locations

Retrieves a list of all locations.

Example Request:

GET /api/locations

Authorization: Bearer <your_token>

5.3 Lane Management

GET /lanes

Retrieves a list of all lanes.

Example Request:

GET /api/lanes

Authorization: Bearer <your_token>

5.4 Device Management

GET /devices

Retrieves a list of all devices.

Example Request:

GET /api/devices

Authorization: Bearer <your_token>

5.5 Access Logs

GET /access-logs

Retrieves a list of access logs with optional filtering.

Example Request:

GET /api/access-logs?start_date=2024-01-01&end_date=2024-01-31&location_id=1

Authorization: Bearer <your_token>

5.6 Vehicle Management

POST /vehicle/presence

Report vehicle presence detection.

Example Request:

POST /api/vehicle/presence

Authorization: Bearer <your_token>

Content-Type: application/json

```
{
    "lane_id": 1,
    "device_id": 1,
    "timestamp": "2024-04-08T10:00:00",
    "confidence": 0.95
}
```

POST /vehicle/anpr

Process ANPR camera result.

Example Request:

POST /api/vehicle/anpr

Authorization: Bearer <your_token>

Content-Type: application/json

```
{
    "lane_id": 1,
    "device_id": 2,
    "vehicle_number": "KA01AB1234",
    "confidence": 0.98,
    "timestamp": "2024-04-08T10:00:01",
    "image_path": "/uploads/anpr/2024/04/08/image_001.jpg"
}
```

5.7 Barrier Management

POST /barrier/control

Control barrier operation.

Example Request:

POST /api/barrier/control

Authorization: Bearer <your_token>

Content-Type: application/json

{
"lane_id": 1,
"device_id": 3,
"action": "open",
"timestamp": "2024-04-08T10:00:02"
}

5.8 Health Management

POST /health/check

Update device health status.

Example Request:

POST /api/health/check

Authorization: Bearer <your_token>

Content-Type: application/json

{
"lane_id": 1,
"device_id": 1,
"device_type": "sensor",
"status": "active",
"last_heartbeat": "2024-04-08T10:00:00",
"error_message": null

}

GET /health/status/{lane_id}

Get health status of all devices in a lane.

Example Request:

GET /api/health/status/1

Authorization: Bearer <your_token>

Example Response:

{
"lane_id": 1,
"lane_name": "Main Gate Lane 1",
"devices": [
{
"device_id": 1,
"device_type": "sensor",
"status": "active",
"last_heartbeat": "2024-04-08T10:00:00"
},
{
"device_id": 2,
"device_type": "anpr",
"status": "active",
"last_heartbeat": "2024-04-08T10:00:00"

```
    }
  ]
}
```

6. Response Codes

200	OK - Request successful
201	Created - Resource created successfully
400	Bad Request - Invalid request parameters
401	Unauthorized - Invalid or missing authentication
403	Forbidden - Insufficient permissions
404	Not Found - Resource not found
500	Internal Server Error - Server error occurred

7. Error Handling

All errors are returned in the following format:

```
{
    "error": {
        "code": "ERROR_CODE",
        "message": "Human readable error message",
        "details": "Additional error details if available"
    }
}
```

8. Rate Limiting

API requests are limited to 1000 requests per hour per API key. Rate limit information is included in the response headers:

X-RateLimit-Limit: 1000

X-RateLimit-Remaining: 999

X-RateLimit-Reset: 1516131940

9. Examples

Python Example:

```
import requests


# Set up the API client
api_key = 'your_api_key'
base_url = 'https://api.onebee.com/v1'
headers = {
    'Authorization': f'Bearer {api_key}',
    'Content-Type': 'application/json'
}


# Get all users
response = requests.get(f'{base_url}/users', headers=headers)
users = response.json()


# Create a new user
new_user = {
    'name': 'John Doe',
    'vehicle_number': 'KA01AB1234',
    'fastag_id': 'FASTAG1234',
    'location_id': 1,
    'valid_from': '2024-01-01',
    'valid_to': '2024-12-31',
```

```
    'is_active': True
}

response = requests.post(f'{base_url}/users', headers=headers, json=new_user)
created_user = response.json()
```