Device App Locations

Endpoints Summary

Method	Path	Swagger
`GET`	`/device_app_locations/`	Swagger ↗

> The Device App Locations API provides comprehensive location management functionality for tracking and organizing physical locations within your organization. This API enables you to retrieve location data with filtering capabilities, manage location hierarchies, and access detailed geographic and organizational information for each location.

Base URL: `https://gate.zequenze.com/api/v1`

Authentication: All endpoints require a Bearer token:

```
Authorization: Bearer <your-api-token>
```

Overview

The Device App Locations API is designed to manage physical locations where devices, equipment, or organizational assets are deployed. This API category serves as the foundation for location-based tracking, reporting, and asset management within the GATE platform.

Key Features:

- Retrieve comprehensive location listings with pagination support
- Filter locations by modification date for synchronization workflows
- Access detailed geographic information including coordinates and address data
- Manage location hierarchies with organizational context
- Support for both active and inactive location tracking

Common Integration Scenarios:

- Asset management systems need to track device locations across multiple facilities
- Inventory applications require location data for warehouse and facility management
- Reporting systems need to generate location-based analytics and summaries
- Mobile applications need to display nearby facilities or service locations
- Synchronization processes need to identify recently updated location records

The API follows REST principles with consistent JSON responses and standard HTTP status codes, making it easy to integrate with existing systems and workflows.

Endpoints

GET /device_app_locations/

Description: Retrieves a paginated list of all locations within your organization. This endpoint provides comprehensive location data including geographic coordinates, address information, organizational context, and metadata. It supports filtering by modification date, making it ideal for synchronization processes and incremental data updates.

Use Cases:

- Populate location dropdowns in device management interfaces
- Synchronize location data with external asset management systems
- Generate location-based reports and analytics dashboards
- Build facility finder applications with geographic search capabilities
- Monitor location configuration changes over time

Full URL Example:

```
https://gate.zequenze.com/api/v1/device_app_locations/?limit=50&offset=0&last_change__gte=2024-01-01T00:00:00Z
```

Parameters:

Parameter	Type	In	Required	Description
last_change__gte	string	query	No	Filter locations modified on or after this date/time (ISO 8601 format). Useful for incremental synchronization

Parameter	Type	In	Required	Description
limit	integer	query	No	Number of results to return per page (default varies, recommended: 10-100)
offset	integer	query	No	The initial index from which to return results for pagination

cURL Example:

```
curl -X GET
"https://gate.zequenze.com/api/v1/device_app_locations/?limit=25&last_change__gte=2024-01-
15T00:00:00Z" \
  -H "Authorization: Bearer YOUR_API_TOKEN" \
  -H "Content-Type: application/json"
```

Example Response:


```
{
  "count": 156,
  "next":
"https://gate.zequenze.com/api/v1/device_app_locations/?limit=25&offset=25&last_change__gte=20
24-01-15T00:00:00Z",
  "previous": null,
  "results": [
    {
      "id": 1001,
      "name": "Downtown Corporate Headquarters",
      "short_name": "downtown-hq",
      "is_active": true,
      "description": "Main corporate office building with executive suites, IT operations
center, and primary data center facilities",
      "organization_id": "org_abc123def456",
      "thumbnail": "https://gate.zequenze.com/media/locations/thumbnails/downtown-hq.jpg",
      "address": "500 Technology Drive, Suite 100",
      "reference": "BLDG-001-HQ",
      "city": "San Francisco",
      "postal_code": "94107",
      "state": "California",
      "region": "Bay Area",
```

```
      "country_code": "US",
      "latitude": "37.7749",
      "longitude": "-122.4194",
      "created": "2023-03-15T09:30:00Z",
      "last_change": "2024-01-20T14:45:33Z"
    },
    {
      "id": 1002,
      "name": "Manufacturing Plant North",
      "short_name": "plant-north",
      "is_active": true,
      "description": "Primary manufacturing facility for hardware production and quality
assurance testing",
      "organization_id": "org_abc123def456",
      "thumbnail": "https://gate.zequenze.com/media/locations/thumbnails/plant-north.jpg",
      "address": "1200 Industrial Boulevard",
      "reference": "MFG-PLANT-N01",
      "city": "Portland",
      "postal_code": "97201",
      "state": "Oregon",
      "region": "Pacific Northwest",
      "country_code": "US",
      "latitude": "45.5152",
      "longitude": "-122.6784",
      "created": "2023-05-22T11:15:00Z",
      "last_change": "2024-01-18T16:22:15Z"
    },
    {
      "id": 1003,
      "name": "Remote Sales Office - Austin",
      "short_name": "sales-austin",
      "is_active": false,
      "description": "Former regional sales office, relocated operations to Dallas facility",
      "organization_id": "org_abc123def456",
      "thumbnail": null,
      "address": "750 Business Center Drive",
      "reference": "SALES-ATX-03",
      "city": "Austin",
      "postal_code": "78759",
      "state": "Texas",
```

```
      "region": "South Central",
      "country_code": "US",
      "latitude": "30.2672",
      "longitude": "-97.7431",
      "created": "2023-01-10T08:00:00Z",
      "last_change": "2024-01-16T10:30:45Z"
    }
  ]
}
```


Response Codes:

Status	Description
200	Success - Returns paginated location data
400	Bad Request - Invalid query parameters (e.g., malformed date)
401	Unauthorized - Invalid or missing API token
403	Forbidden - Insufficient permissions to access location data
429	Too Many Requests - Rate limit exceeded
500	Internal Server Error - Server-side processing error

Common Use Cases

Use Case 1: Device Assignment Dashboard

Build an interface where administrators can assign devices to specific locations. Use this endpoint to populate location dropdown menus with active facilities, showing both the full name for clarity and short_name for compact displays.

Use Case 2: Incremental Data Synchronization

Implement a scheduled sync process that periodically checks for location updates using the `last_change__gte` parameter. This enables efficient data synchronization without downloading the entire location dataset each time.

Use Case 3: Geographic Asset Mapping

Create interactive maps showing asset distribution across facilities using the latitude/longitude coordinates. Combine location data with device counts to visualize organizational presence and asset density.

Use Case 4: Location-Based Reporting

Generate facility utilization reports by filtering locations based on active status and grouping by region or state. Use the reference codes for consistent identification across different reporting systems.

Use Case 5: Mobile Field Service Applications

Build mobile apps that help technicians find nearby service locations using geographic coordinates. Filter for active locations only and use thumbnail images to help with visual identification of facilities.

Best Practices

Pagination Management:

- Use reasonable page sizes (25-100 records) to balance performance and network efficiency
- Always handle the `next` and `previous` URLs for navigation rather than manually calculating offsets
- Monitor the `count` field to understand total dataset size for progress indicators

Efficient Data Synchronization:

- Implement incremental sync using `last_change__gte` with the timestamp from your last successful sync
- Store the `last_change` timestamp from the most recently updated record for the next sync cycle
- Consider implementing exponential backoff for failed sync attempts

Geographic Data Handling:

- Validate latitude/longitude coordinates before using them in mapping applications
- Handle cases where geographic coordinates might be null for some locations
- Consider implementing geocoding fallbacks using the address fields when coordinates are missing

Performance Optimization:

- Cache frequently accessed location data locally with appropriate TTL based on your update frequency
- Use the `is_active` field to filter out decommissioned locations in user-facing interfaces
- Index location data by commonly filtered fields like `short_name` or `reference` for faster lookups

Error Handling:

- Implement proper retry logic for transient errors (429, 500 status codes)
- Validate date parameters client-side before sending requests to avoid 400 errors
- Handle cases where locations might be deleted between API calls (404 responses)

Revision #4
Created 2026-02-04 05:13:41 UTC by ipena@zequenze.com
Updated 2026-02-11 03:14:56 UTC by ipena@zequenze.com