

Sisukord

Creating a POSTMAN collection file

3

Importing the file into Postman

86

Creating a POSTMAN collection file

Download file [API json file](#) by left click and Save As . . it

OR

Open Notepad (Start → Notepad) and paste in the code below:

```
{
  "info": {
    "_postman_id": "6de645db-4111-4d25-aeb3-016b7bb4c0cf",
    "name": "Directo API",
    "description": "# Directo API Documentation\n\nWelcome to the Directo API - a comprehensive RESTful API for accessing and managing your ERP data including items, orders, customers, invoices, stock levels, and more.\n\n---\n\n## Getting Started\n\n### Prerequisites\n\n- Active Directo ERP account\n  \n- API key (obtain from your Directo administrator)\n  \n- Basic understanding of REST APIs and HTTP methods\n  \n\n### Quick Start Guide\n\n1. **Configure Authentication**: Set your API key in the collection's Authorization tab or update the `X-Directo-Key` header\n\n2. **Set Variables**: Configure `baseUrl` and `version` in collection variables\n\n3. **Choose Format**: Set the `Accept` header to `application/json` or `application/xml`\n\n4. **Make Your First Request**: Try the `\"All items\"` request to retrieve your item catalog\n\n\n### Base URL & Versioning\n\n**Base URL Structure**: `{{baseUrl}}/v{{version}}/{resource}`\n\n\n**Current Configuration**:\n\n- `baseUrl`: `https://login.directo.ee/apidirect`\n  \n- `version`: `1` (current API version)\n  \n\n**Example Full URL**: `https://login.directo.ee/apidirect/v1/items`\n\n\nThe API uses versioning to ensure backward compatibility. Always specify the version number in your requests.\n\n---\n\n## Authentication\n\nThe Directo API uses **API Key authentication** for secure access.\n\n### How to Authenticate\n\nInclude your API key in the request header:\n\n`` http\nX-Directo-Key: YOUR_API_KEY_HERE\n\n\n### Getting Your API Key\n\nContact your Directo system administrator to obtain an API key. Each key is associated with specific permissions and access levels.\n\n### Security Best Practices\n\n- **Never expose your API key** in client-side code or public repositories\n  \n- Store API keys securely using environment variables\n  \n- Rotate keys periodically\n  \n- Use different keys for development and production environments\n  \n\n---\n\n## Response Formats (JSON vs XML)\n\nThe API supports both JSON and XML response formats. Control the format using the `Accept` header.\n\n### JSON Format (Recommended)\n\n`` http\nAccept: application/json\n\n\n**Example Request**:\n\n`` bash\ncurl -X GET \"{{baseUrl}}/v{{version}}/items\" \\n -H \"X-Directo-Key: YOUR_API_KEY\\n\\n -H \"Accept: application/json\"\n\n\n### XML Format\n\n`` http\nAccept: application/xml\n\n\n**Example Request**:\n\n`` bash\ncurl -X GET \"{{baseUrl}}/v{{version}}/items\" \\n -H \"X-Directo-
```

```
Key: YOUR_API_KEY"\n\n -H \"Accept: application/xml\"\n\n ``\n\n---\n\n\n## □ Advanced Filtering (Key Feature)\n\n\n### □ ERP-Style Filtering - The Power of Directo API\n\n\n**□ CRITICAL FEATURE: The Directo API supports the exact same powerful filtering capabilities as the Directo ERP system itself.**\n\n\nThis is one of the most powerful features of the Directo API. **Filters can be applied like default filters inside the ERP system. All fields can be used for filters except calculated fields.**\n\n\n#### What This Means for Developers\n\n\n1. **Seamless Integration**: Any filter that works in your Directo ERP interface can be applied via the API using query parameters\n\n\n2. **No Learning Curve**: If you know how to filter in the ERP, you already know how to filter via the API\n\n\n3. **Maximum Flexibility**: Filter by ANY field in the response data (except calculated fields)\n\n\n4. **Powerful Queries**: Combine multiple filters to create complex queries without custom endpoints\n\n\n\n##### Understanding Filterable vs Calculated Fields\n\n\n**□ Filterable Fields (Stored in Database)** \n\nThese are fields that are stored directly in the database and can be used for filtering:\n\n\n- **Item Fields**: `code`, `name`, `class`, `status`, `country`, `unit`, `price`, `cost`, `supplier_code`, `barcode`, `weight`, `volume`\n\n\n- **Order/Invoice Fields**: `number`, `date`, `status`, `customer_code`, `customer_name`, `total`, `currency`, `payment_terms`, `delivery_date`, `warehouse`\n\n\n- **Customer Fields**: `code`, `name`, `country`, `city`, `address`, `phone`, `email`, `vat_number`, `payment_terms`, `credit_limit`\n\n\n- **Stock Fields**: `item_code`, `warehouse`, `quantity`, `reserved_quantity`, `available_quantity`, `location`, `batch_number`, `serial_number`\n\n\n- **Line Item Fields**: `row_item`, `row_quantity`, `row_price`, `row_description`, `row_discount`, `row_warehouse`\n\n\n- **Custom Data Fields**: Any custom fields you've defined in your ERP system (found in `datafields` array)\n\n\n\n**□ Calculated Fields (Cannot Be Filtered)** \n\nThese are fields computed on-the-fly and cannot be used for filtering:\n\n\n- **Computed Totals**: Fields like `total_with_vat`, `line_total`, `discount_amount` (when calculated from other fields)\n\n\n- **Derived Values**: `available_stock` (when calculated as `quantity - reserved`), `profit_margin` (calculated from price and cost)\n\n\n- **Aggregations**: `total_orders`, `average_price`, `sum_quantity` (when aggregated from multiple records)\n\n\n- **Formatted Values**: Display-only fields like `formatted_date`, `formatted_price` (when they're just formatting of other fields)\n\n\n- **Virtual Fields**: Fields that exist only in the API response but not in the database\n\n\n\n##### How to Identify Calculated Fields\n\n\n1. **Check the ERP Interface**: If you cannot filter by a field in the ERP system's filter dialog, it's likely calculated\n\n\n2. **Test the Filter**: Try filtering by the field via API - if it returns no results or an error, it's calculated\n\n\n3. **Look for Patterns**: Fields with names like `total_`, `calculated_`, `formatted_`, `display_` are often calculated\n\n\n4. **Consult Documentation**: Check your ERP
```

system's field documentation or ask your administrator\n \n\n####

Practical Examples: Filterable vs Calculated\n\n**Example 1: Item Stock**\n\n`` http\nWORKS: GET /items?quantity=>100\n (quantity is stored in database)\n DOESN'T WORK: GET /items?available\_stock=>100\n (if available\_stock = quantity - reserved, it's calculated)\n\n``\n\n**Example 2: Order Totals**\n\n`` http\nWORKS: GET /orders?row_price=>1000\n (row_price is stored per line item)\n MIGHT NOT WORK: GET /orders?order_total=>5000\n (if order_total is sum of all line items, it's calculated)\n ALTERNATIVE: GET /orders?date=>2025-01-01&status=COMPLETED\n (then calculate totals in your application)\n\n``\n\n\*\*Example 3: Customer Data\*\*\n\n`` http\nWORKS: GET /customers?country=EE&city=Tallinn\n (both are stored fields)\n DOESN'T WORK: GET /customers?full_address=*Tallinn*\n (if full_address is concatenation of address+city+country)\n ALTERNATIVE: GET /customers?city=Tallinn\n (filter by individual stored fields)\n\n``\n\n#### Basic Filtering\n\nFilter by exact match using any stored field name:\n\n`` http\nGET {{baseUrl}}/v{{version}}/items?code=9810\nGET {{baseUrl}}/v{{version}}/items?status=UUS\nGET {{baseUrl}}/v{{version}}/orders?number=12345\nGET {{baseUrl}}/v{{version}}/customers?country=EE\nGET {{baseUrl}}/v{{version}}/items?class=KAUP\n\n``\n\n**Key Principle**: Use the exact field name as it appears in the API response (or as you see it in the ERP system).\n\n#### Contains Filtering\n\nPartial filtering allows you to search for values **containing** a specific substring.\n\nThis works like SQL `LIKE '%text%'` logic.\n\nUse it when you need flexible filtering for names, codes, descriptions, or any other text fields\n\n``\nGET {{baseUrl}}/v{{version}}/customers?name=%john%\n\n``\n\n#### Comparison Operators\n\nThe API supports powerful comparison operators for advanced filtering:\n\n#### Greater Than (>)\n\n`` http\nGET {{baseUrl}}/v{{version}}/orders?date=>2025-09-01T00:00:00\nGET {{baseUrl}}/v{{version}}/invoices?row\_quantity=>5000\n\n``\n\n#### Less Than (<)\n\n`` http\nGET {{baseUrl}}/v{{version}}/orders?date=<2025-09-03T00:00:00\nGET {{baseUrl}}/v{{version}}/invoices?row\_quantity=<5000\n\n``\n\n#### Not Equal (!)\n\n`` http\nGET {{baseUrl}}/v{{version}}/items?country=!EE\nGET {{baseUrl}}/v{{version}}/orders?status=!CLOSED\nGET {{baseUrl}}/v{{version}}/customers?payment\_terms=!COD\n\n``\n\n#### Greater Than or Equal (>=)\n\n`` http\nGET {{baseUrl}}/v{{version}}/items?price=>100\nGET {{baseUrl}}/v{{version}}/orders?date=>2025-01-01T00:00:00\n\n``\n\n#### Less Than or Equal (<=)\n\n`` http\nGET {{baseUrl}}/v{{version}}/items?weight=<=5.5\nGET {{baseUrl}}/v{{version}}/orders?date=<2025-12-31T23:59:59\n\n``\n\n#### Range Filtering\n\nCombine operators to create ranges (AND logic):\n\n`` http\nGET {{baseUrl}}/v{{version}}/orders?date=>2025-09-01T00:00:00&date=<2025-09-03T00:00:00\nGET {{baseUrl}}/v{{version}}/items?quantity=>10&quantity=<100\nGET {{baseUrl}}/v{{version}}/invoices?total=>1000&total=<5000\nGET {{baseUrl}}/v{{version}}/items?price=>50&price=<=200\n\n``\n\n#### Multiple Values (OR Logic)\n\nUse comma-separated lists to filter by multiple values:\n\n`` http\nGET

```
{{baseUrl}}/v{{version}}/items?warehouse=AALTR,AIPM,WAREHOUSE1\nGET
{{baseUrl}}/v{{version}}/orders?status=NEW,PENDING,PROCESSING\nGET
{{baseUrl}}/v{{version}}/customers?country=EE,LV,LT\nGET
{{baseUrl}}/v{{version}}/items?class=KAUP,TEENUS,TOORAINEN\n\n```\n\n###
Timestamp-Based Filtering (Synchronization)\n\nUse the `ts` parameter to
retrieve records modified after a specific timestamp (ideal for
synchronization):\n\n```\nhttp\nGET
{{baseUrl}}/v{{version}}/items?ts=>2025-10-01T08:11:05.129Z\nGET
{{baseUrl}}/v{{version}}/customers?ts=>2025-01-01T00:00:00Z\nGET
{{baseUrl}}/v{{version}}/orders?ts=>2025-11-20T07:30:00Z\n\n```\n\n**Best
Practice**: Store the timestamp of your last successful sync and use it in
the next request to fetch only changed records.\n\n### Filtering by Line
Item Fields\n\nFor resources with line items (orders, invoices), you can
filter by line-level fields:\n\n```\nhttp\nGET
{{baseUrl}}/v{{version}}/orders?row_item=343443\nGET
{{baseUrl}}/v{{version}}/invoices?row_quantity=>10\nGET
{{baseUrl}}/v{{version}}/orders?row_description=Maja\nGET
{{baseUrl}}/v{{version}}/priceformulas?row_item=343443\n\n```\n\n**Note**:
When filtering by line item fields, the API returns the entire document
(order/invoice) if ANY line matches the filter.\n\n### Filtering by Custom
Data Fields\n\nIf you have custom fields defined in your ERP system (visible
in the `datafields` array), you can filter by them:\n\n```\nhttp\nGET
{{baseUrl}}/v{{version}}/items?data_row_code=BRAND\nGET
{{baseUrl}}/v{{version}}/items?data_row_code=ART_PROJ,BRAND\nGET
{{baseUrl}}/v{{version}}/items?data_row_content=0\n\n```\n\n**How to Find
Custom Field Names**: Make a request without filters and examine the
`datafields` array in the response.\n\n### Complex Filter
Combinations\n\nCombine multiple filters to create sophisticated
queries:\n\n**Get new orders from the last week for a specific
customer:**\n\n```\nhttp\nGET
{{baseUrl}}/v{{version}}/orders?status=NEW&date=>2025-11-13T00:00:00&custome
r_code=CUST001\n\n```\n\n**Get items with low stock in specific warehouses,
excluding closed items:**\n\n```\nhttp\nGET
{{baseUrl}}/v{{version}}/items?quantity=<10&warehouse=AALTR,AIPM&status!=CLO
SED\n\n```\n\n**Get all customers from Baltic countries with high credit
limits:**\n\n```\nhttp\nGET
{{baseUrl}}/v{{version}}/customers?country=EE,LV,LT&credit_limit=>50000\n\n
```\n\n**Get invoices from Q1 2025 with specific payment terms:**\n\n```\n
http\nGET
{{baseUrl}}/v{{version}}/invoices?date=>2025-01-01T00:00:00&date=<2025-04-01
T00:00:00&payment_terms=NET30\n\n```\n\n**Get items from specific class
with price range:**\n\n```\nhttp\nGET
{{baseUrl}}/v{{version}}/items?class=KAUP&price=>10&price=<100\n\n
```\n\n### Common Filter Parameters by Resource\n\n| Parameter | Description
| Example | Applicable Resources |\n| --- | --- | --- | --- |\n| `ts` |
Timestamp for modified records (ISO 8601) | `2025-10-01T08:11:05.129Z` | All
resources |\n| `date` | Date filtering (supports operators) |
`>2025-09-01T00:00:00` | Orders, Invoices |\n| `number` | Document number |
`12345` or `!001` | Orders, Invoices |\n| `code` | Item/Customer code |
`9810` or `CUST001` | Items, Customers |\n| `name` | Name search | `Kalev` |
Items, Customers |\n| `row_item` | Item code in line items | `9810` or
```

```

`PROD-ABC` | Orders, Invoices, Price Formulas | \n| `status` | Status filter
| `UUS`, `CLOSED`, `NEW` | Orders, Items | \n| `country` | Country code (ISO
2-letter) | `EE`, `!EE` | Items, Customers | \n| `warehouse` |
Warehouse/stock location code | `AALTR,AIPM` | Items, Stock Levels, Orders
| \n| `quantity` | Quantity filter (supports operators) | `>100`, `<50` |
Items, Stock Levels | \n| `row_quantity` | Line item quantity (supports
operators) | `>10` | Invoices, Orders | \n| `row_description` | Line item
description text | `Maja` | Invoices, Orders | \n| `closed` | Closed status |
`1` (closed), `0` (open) | Customers, Orders | \n| `class` | Item
class/category | `KAUP`, `TEENUS` | Items | \n| `price` | Price filter
(supports operators) | `>100`, `<=500` | Items | \n| `customer_code` |
Customer identifier | `CUST001` | Orders, Invoices | \n\n#### Filter Best
Practices\n\n1. **Start Simple**: Begin with single filters and add
complexity as needed\n    \n2. **Use Timestamp Sync**: Always use `ts`
parameter for incremental synchronization\n    \n3. **Combine
Strategically**: Use multiple filters to reduce data transfer and
processing\n    \n4. **Test in ERP First**: If unsure whether a field is
filterable, test it in the ERP interface first\n    \n5. **Handle Empty
Results**: Empty results might mean filters are too restrictive or field is
calculated\n    \n6. **URL Encode**: Always URL-encode filter values,
especially dates and special characters\n    \n7. **Document Your Filters**:
Keep track of which filters work for your use cases\n    \n\n####
Troubleshooting Filters\n\n**Problem**: Filter returns no results\n\n-
**Check**: Is the field calculated? Try filtering by underlying stored
fields\n    \n- **Check**: Is the value correct? Verify exact field values
in an unfiltered response\n    \n- **Check**: Are operators correct? Use `>`
not `>=` if that's what the ERP uses\n    \n\n**Problem**: Filter seems to
be ignored\n\n- **Check**: Is the field name spelled correctly? Field names
are case-sensitive\n    \n- **Check**: Is the value URL-encoded? Special
characters must be encoded\n    \n- **Check**: Does the field exist in this
resource? Not all fields are available in all resources\n\n\n**Problem**: Unexpected results\n\n- **Check**: Are you using OR logic
(comma) when you meant AND (multiple parameters)?\n    \n- **Check**: For
line item filters, remember the entire document is returned if any line
matches\n    \n\n---\n\n## API Resources & Endpoints\n\nAPI Resources and
Endpoint in this section are samples and doesn't reflect all available
resources. View Postman collection data to get full list.\n\n\nResponse field
list managed in Directo ERP.\n\n#### Items\n\nManage your product catalog and
inventory items.\n\n**Endpoint:** `GET
{{baseUrl}}/v{{version}}/items`\n\n**Description**: Retrieve item master
data including product codes, names, classifications, pricing, and custom
data fields.\n\n**Use Cases:**\n\n- Retrieve product catalog for e-commerce
integration\n    \n- Sync inventory data to external systems\n    \n- Filter
items by stock location, quantity, or status\n    \n- Export product data
for reporting\n    \n\n**Common Filters:** `code`, `name`, `class`,
`status`, `warehouse`, `quantity`, `country`, `price`, `ts`\n\n**Example
Requests:**\n\n`` http\n# Get all active items\nGET
{{baseUrl}}/v{{version}}/items?status=ACTIVE\n# Get items with low
stock\nGET {{baseUrl}}/v{{version}}/stocklevels?quantity=<10\n# Get items
modified since last sync\nGET
{{baseUrl}}/v{{version}}/items?ts=2025-11-20T07:00:00Z\n# Get items in

```

```

specific warehouses\nGET
{{baseUrl}}/v{{version}}/items?warehouse=AALTR,AIPM\n\n ``\n\n**Response
Fields:**\n\n- `code`: Item code/SKU\n    \n- `name`: Item
name/description\n    \n- `class`: Item classification\n    \n- `status`:
Item status\n    \n- `price`: Unit price\n    \n- `cost`: Unit cost\n    \n-
`quantity`: Stock quantity\n    \n- `warehouse`: Warehouse location\n    \n-
`datafields`: Array of custom data fields\n    \n\n---\n\n###
Customers\n\nAccess customer and client information.\n\n**Endpoint:** `GET
{{baseUrl}}/v{{version}}/customers`\n\n**Description:** Retrieve customer
master data including contact information, addresses, payment terms, and
credit limits.\n\n**Use Cases:**\n\n- Retrieve customer database for CRM
integration\n    \n- Sync customer data to external systems\n    \n- Filter
by country or modification date\n    \n- Export customer lists for
marketing\n    \n\n**Common Filters:** `code`, `name`, `country`, `city`,
`payment_terms`, `credit_limit`, `closed`, `ts`\n\n**Example
Requests:**\n\n`` http\n# Get all customers from Estonia\nGET
{{baseUrl}}/v{{version}}/customers?country=EE\n# Get customers with high
credit limits\nGET {{baseUrl}}/v{{version}}/customers?credit_limit=>50000\n#
Get active customers only\nGET
{{baseUrl}}/v{{version}}/customers?closed=0\n# Get customers modified since
last sync\nGET
{{baseUrl}}/v{{version}}/customers?ts=2025-11-20T07:00:00Z\n\n
``\n\n**Response Fields:**\n\n- `code`: Customer code\n    \n- `name`:
Customer name\n    \n- `country`: Country code (ISO 2-letter)\n    \n-
`city`: City\n    \n- `address`: Street address\n    \n- `phone`: Phone
number\n    \n- `email`: Email address\n    \n- `vat_number`: VAT
registration number\n    \n- `payment_terms`: Payment terms code\n    \n-
`credit_limit`: Credit limit amount\n    \n- `closed`: Closed status
(0=active, 1=closed)\n    \n\n---\n\n### Orders\n\nRetrieve sales orders and
purchase orders.\n\n**Endpoint:** `GET
{{baseUrl}}/v{{version}}/orders`\n\n**Description:** Retrieve order
documents including headers and line items. Supports filtering by date
ranges, status, customer, and line item details.\n\n**Use Cases:**\n\n-
Retrieve order history for reporting\n    \n- Monitor order status for
fulfillment\n    \n- Filter by date range, status, or customer\n    \n- Sync
orders to external fulfillment systems\n    \n- Track orders with specific
items\n    \n\n**Common Filters:** `number`, `date`, `status`,
`customer_code`, `warehouse`, `closed`, `row_item`, `row_quantity`,
`ts`\n\n**Example Requests:**\n\n`` http\n# Get orders from date range\nGET
{{baseUrl}}/v{{version}}/orders?date=>2025-09-01T00:00:00&date=<2025-09-03T0
0:00:00\n# Get new orders\nGET {{baseUrl}}/v{{version}}/orders?status=NEW\n#
Get orders containing specific item\nGET
{{baseUrl}}/v{{version}}/orders?row_item=343443\n# Get orders for specific
customer\nGET {{baseUrl}}/v{{version}}/orders?customer_code=CUST001\n# Get
open orders modified today\nGET
{{baseUrl}}/v{{version}}/orders?closed=0&ts=2025-11-20T00:00:00Z\n\n
``\n\n**Response Fields:**\n\n- `number`: Order number\n    \n- `date`:
Order date\n    \n- `status`: Order status\n    \n- `customer_code`:
Customer code\n    \n- `customer_name`: Customer name\n    \n- `warehouse`:
Warehouse code\n    \n- `total`: Order total amount\n    \n- `currency`:
Currency code\n    \n- `closed`: Closed status\n    \n- `lines`: Array of

```

```

order line items\n      \n      - `row_item`: Item code\n      \n      -
`row_quantity`: Quantity\n      \n      - `row_price`: Unit price\n
\n      - `row_description`: Line description\n      \n\n---\n\n###
Invoices\n\nAccess invoice data and billing information.\n\n**Endpoint:**
`GET {{baseUrl}}/v{{version}}/invoices`\n\n**Description**: Retrieve invoice
documents including headers and line items. Supports filtering by date
ranges, status, customer, and line item details.\n\n**Use Cases:**\n\n-
Retrieve invoice history for accounting\n      \n- Export billing data to
accounting systems\n      \n- Filter by date, status, or customer\n      \n-
Track invoices with specific items\n      \n- Monitor payment status\n
\n\n**Common Filters:** `number`, `date`, `status`, `customer_code`,
`total`, `closed`, `row_item`, `row_quantity`, `ts`\n\n**Example
Requests:**\n\n`` http\n\n# Get invoices from last month\n\nGET
{{baseUrl}}/v{{version}}/invoices?date=>2025-10-01T00:00:00&date=<2025-11-01
T00:00:00\n\n# Get unpaid invoices\n\nGET
{{baseUrl}}/v{{version}}/invoices?closed=0\n\n# Get invoices containing
specific item\n\nGET {{baseUrl}}/v{{version}}/invoices?row_item=9810\n\n
``\n\n**Response Fields:**\n\n- `number`: Invoice number\n      \n- `date`:
Invoice date\n      \n- `due_date`: Payment due date\n      \n- `status`:
Invoice status\n      \n- `customer_code`: Customer code\n      \n-
`customer_name`: Customer name\n      \n- `total`: Invoice total amount\n
\n- `currency`: Currency code\n      \n- `closed`: Payment status\n      \n-
`lines`: Array of invoice line items\n      \n\n---\n\n### Stock
Levels\n\nMonitor inventory levels across warehouses.\n\n**Endpoint:** `GET
{{baseUrl}}/v{{version}}/stocklevels`\n\n**Description**: Retrieve current
stock levels for items across all warehouses. Shows quantity on hand,
reserved quantity, and available quantity.\n\n**Use Cases:**\n\n- Check
current inventory levels\n      \n- Monitor stock across locations\n      \n-
Identify low-stock items\n      \n- Sync inventory to e-commerce platforms\n
\n- Calculate available-to-promise\n      \n\n**Common Filters:** `item_code`,
`warehouse`, `quantity`, `reserved_quantity`, `ts`\n\n**Example
Requests:**\n\n`` http\n\n# Get stock levels for specific item\n\nGET
{{baseUrl}}/v{{version}}/stocklevels?item_code=9810\n\n# Get low stock
items\n\nGET {{baseUrl}}/v{{version}}/stocklevels?quantity=<10\n\n# Get stock in
specific warehouse\n\nGET
{{baseUrl}}/v{{version}}/stocklevels?warehouse=AALTR\n\n# Get stock changes
since last sync\n\nGET
{{baseUrl}}/v{{version}}/stocklevels?ts=2025-11-20T07:00:00Z\n\n
``\n\n**Response Fields:**\n\n- `item_code`: Item code\n      \n-
`warehouse`: Warehouse code\n      \n- `quantity`: Quantity on hand\n      \n-
`reserved_quantity`: Reserved/allocated quantity\n      \n-
`available_quantity`: Available quantity (may be calculated)\n      \n-
`location`: Storage location within warehouse\n      \n\n---\n\n### Stock
Levels with Serial/Batch Numbers\n\nRetrieve detailed stock information
including serial numbers and batch numbers.\n\n**Endpoint:** `GET
{{baseUrl}}/v{{version}}/stocklevels_sn`\n\n**Description**: Retrieve stock
levels with serial number and batch number tracking. Essential for lot-
tracked and serialized inventory.\n\n**Use Cases:**\n\n- Track serialized
inventory (electronics, equipment)\n      \n- Manage batch-tracked items
(food, pharmaceuticals)\n      \n- Trace product lots for recalls\n      \n-
Monitor expiration dates by batch\n      \n- Audit serial number assignments\n

```

```

\n\n**Common Filters:** `item_code`, `warehouse`, `serial_number`,
`batch_number`, `ts`\n\n**Example Requests:**\n\n`` http\n# Get serialized
stock for specific item\nGET
{{baseUrl}}/v{{version}}/stocklevels_sn?item_code=LAPTOP001\n# Get stock by
batch number\nGET
{{baseUrl}}/v{{version}}/stocklevels_sn?batch_number=BATCH2025001\n# Get
serialized stock in warehouse\nGET
{{baseUrl}}/v{{version}}/stocklevels_sn?warehouse=AALTR\n\n
``\n\n**Response Fields:**\n\n- `item_code`: Item code\n    \n-
`warehouse`: Warehouse code\n    \n- `serial_number`: Serial number (for
serialized items) or Batch/lot number (for batch-tracked items)\n    \n-
`quantity`: Quantity (typically 1 for serialized items)\n    \n-
`expiration_date`: Expiration date (for batch-tracked items)\n    \n\n---
\n\n### Allocations\n\nView inventory allocations and
reservations.\n\n**Endpoint:** `GET
{{baseUrl}}/v{{version}}/allocations`\n\n**Description**: Retrieve inventory
allocations showing which items are reserved for specific orders or
purposes.\n\n**Use Cases:**\n\n- Check reserved inventory\n    \n- Monitor
order allocations\n    \n- Prevent overselling\n    \n- Calculate available-
to-promise\n    \n- Track allocation by order\n    \n\n**Common Filters:**
`code`, `warehouse`, `ts`\n\n**Example Requests:**\n\n`` http\n# Get
allocations for specific item\nGET
{{baseUrl}}/v{{version}}/allocations?code=9810\n# Get allocations in
warehouse\nGET {{baseUrl}}/v{{version}}/allocations?warehouse=AALTR\n\n
``\n\n**Response Fields:**\n\n- `item_code`: Item code\n    \n-
`warehouse`: Warehouse code\n    \n- `quantity`: Allocated quantity\n
\n\n---\n\n### Item Classes\n\nRetrieve item classification and category
data.\n\n**Endpoint:** `GET
{{baseUrl}}/v{{version}}/itemclasses`\n\n**Description**: Retrieve item
class/category master data used to classify and organize items in the ERP
system.\n\n**Use Cases:**\n\n- Get product categories for navigation\n
\n- Sync classification hierarchies\n    \n- Filter items by class\n    \n-
Build category trees for e-commerce\n    \n\n**Common Filters:** `code`,
`name`, `parent_code`, `ts`\n\n**Example Requests:**\n\n`` http\n# Get all
item classes\nGET {{baseUrl}}/v{{version}}/itemclasses\n# Get classes
modified since last sync\nGET
{{baseUrl}}/v{{version}}/itemclasses?ts=2025-11-20T07:00:00Z\n\n
``\n\n**Response Fields:**\n\n- `code`: Class code\n    \n- `name`: Class
name\n    \n- `parent_code`: Parent class code (for hierarchies)\n    \n-
`description`: Class description\n    \n\n---\n\n### Price
Formulas\n\nAccess pricing rules and formulas.\n\n**Endpoint:** `GET
{{baseUrl}}/v{{version}}/priceformulas`\n\n**Description**: Retrieve pricing
formulas and rules used to calculate dynamic prices based on various
conditions.\n\n**Use Cases:**\n\n- Retrieve pricing rules for specific
items\n    \n- Calculate dynamic prices in external systems\n    \n- Sync
pricing logic\n    \n- Audit pricing formulas\n    \n\n**Common Filters:**
`row_item`, `customer_code`, `ts`\n\n**Example Requests:**\n\n`` http\n#
Get price formulas for specific item\nGET
{{baseUrl}}/v{{version}}/priceformulas?row_item=343443\n# Get all price
formulas\nGET {{baseUrl}}/v{{version}}/priceformulas\n# Get formulas
modified since last sync\nGET

```

```

{{baseUrl}}/v{{version}}/priceformulas?ts=2025-11-20T07:00:00Z\n\n
``\n\nResponse Fields:\n\n- `row_item`: Item code\n    \n-
`customer_code`: Customer code (if customer-specific)\n    \n- `formula`:
Pricing formula\n    \n- `discount_percent`: Discount percentage\n    \n-
`valid_from`: Valid from date\n    \n- `valid_to`: Valid to date\n    \n\n-
- \n\nDeletions\n\nTrack deleted records for synchronization
purposes.\n\nEndpoint: `GET`
{{baseUrl}}/v{{version}}/deleted\n\nDescription: Retrieve records that
have been deleted from the ERP system. Essential for maintaining data
consistency in external systems.\n\nUse Cases:\n\n- Sync deletions to
external systems\n    \n- Maintain data consistency\n    \n- Audit deleted
records\n    \n- Clean up orphaned references\n    \n\nCommon Filters:
`resource_type`, `ts`\n\nExample Requests:\n\n`` http\n# Get all
deletions since last sync\nGET
{{baseUrl}}/v{{version}}/deleted?ts=2025-11-20T07:00:00Z\n# Get deleted
items only\nGET {{baseUrl}}/v{{version}}/deleted?resource_type=items\n\n
``\n\nResponse Fields:\n\n- `resource_type`: Type of deleted resource
(items, customers, orders, etc.)\n    \n- `resource_id`: ID of deleted
resource\n    \n- `deleted_at`: Deletion timestamp\n    \n\n- - - \n\n
Response Structure\n\nSuccess Response (HTTP 200)\n\nJSON
Format:\n\n`` json\n[\n    {\n        "code": "1011",\n
        "class": "KAUP",\n        "name": "Kommid Lily, Kalev",\n
        "price": 12.50,\n        "quantity": 150,\n        "warehouse":
        "AALTR",\n        "datafields": [\n            {\n
                "code": "TEST_ARTIKKEL",\n                "content": "Test lisaväli
sisu"\n            },\n            {\n                "code":
                "IS_WEIGHT",\n                "content": "0"\n            },\n
            {\n                "code": "VISUAL_VERIFICATION",\n
                "content": "0"\n            },\n            {\n
                "code": "VENSAFE",\n                "content": "0"\n            }
        ],\n        {\n            "code": "1012",\n            "class": "KAUP",\n
            "name": "Šokolaad Kalev",\n            "price": 8.75,\n
            "quantity": 200,\n            "warehouse": "AIPM",\n
            "datafields": []\n        }\n    ]\n\n``\n\nXML Format:\n\n``
xml\n<items>\n    <item code="1011" class="KAUP" name="Kommid Lily,
Kalev" price="12.50" quantity="150" warehouse="AALTR">\n
        <datafields>\n            <data code="TEST_ARTIKKEL" content="Test
lisaväli sisu"/>\n            <data code="IS_WEIGHT" content="0"/>\n
            <data code="VISUAL_VERIFICATION" content="0"/>\n            <data
code="VENSAFE" content="0"/>\n        </datafields>\n    </item>\n
    <item code="1012" class="KAUP" name="Šokolaad Kalev" price="8.75"
quantity="200" warehouse="AIPM">\n        <datafields/>\n
    </item>\n</items>\n\n``\n\nResponse Characteristics\n\n- Array
Format: All endpoints return arrays of objects, even for single results\n
- Consistent Structure: Field names are consistent across all response
formats\n    \n- Custom Fields: Custom data fields appear in the
`datafields` array\n    \n- Null Values: Missing or null values may be
omitted from responses\n    \n- Date Format: All dates use ISO 8601
format (e.g., `2025-11-20T07:30:00Z`)\n    \n- Numeric Values: Numbers
are returned as numbers in JSON, as strings in XML\n    \n\n- - - \n\n
Error Handling\n\nHTTP Status Codes\n\n| Status Code | Meaning |

```

Description | \n | --- | --- | --- | \n | `200` | OK | Request successful | \n | `400` | Bad Request | Invalid parameters or malformed request | \n | `401` | API not enabled | Resource not enabled for this API key | \n | `403` | Unauthorized | Missing or invalid API key | \n | `404` | Forbidden | API key lacks required permissions | \n | `409` | Not Found | Resource or endpoint not found | \n | `429` | Too Many Requests | Rate limit exceeded | \n | `500` | Internal Server Error | Server-side error | \n | `503` | Service Unavailable | API temporarily unavailable | \n \n ### Error Response Format \n \n `` json \n { \n "error": { \n "code": "INVALID\_API\_KEY", \n "message": "The provided API key is invalid or expired", \n "details": "Please check your API key and try again" \n } \n } \n \n `` \n \n ### Common Error Scenarios \n \n **401 Unauthorized:** \n \n - Missing `X-Directo-Key` header \n - Invalid or expired API key \n - **Solution:** Verify your API key and ensure it's included in the header \n \n **403 Forbidden:** \n \n - API key lacks permissions for the requested resource \n - **Solution:** Contact your administrator to grant necessary permissions \n \n **400 Bad Request:** \n \n - Invalid filter syntax \n - Malformed date format (use ISO 8601: `2025-11-20T07:30:00Z`) \n - Filtering by calculated field \n - **Solution:** Check filter parameters and date formats \n \n **404 Not Found:** \n \n - Incorrect endpoint URL \n - Invalid API version \n - **Solution:** Verify the endpoint path and version number \n \n **429 Too Many Requests:** \n \n - Rate limit exceeded \n - **Solution:** Implement exponential backoff and reduce request frequency \n \n **500 Internal Server Error:** \n \n - Server-side processing error \n - **Solution:** Retry after a delay; contact support if persistent \n \n - \n \n ## ☐ Rate Limiting & Performance \n \n ### Rate Limits \n \n While specific rate limits depend on your account tier, follow these guidelines: \n \n - **Recommended:** Max 60 requests per minute \n - **Burst Limit:** Short bursts of higher traffic may be allowed \n - **Daily Limits:** May apply based on your subscription \n \n \n ### Performance Best Practices \n \n \n #### 1 \n \n Use Timestamp Filtering for Sync \n \n Instead of fetching all records repeatedly, use the `ts` parameter: \n \n `` http \n GET {{baseUrl}}/v{{version}}/items?ts=>2025-11-20T07:00:00Z \n \n `` \n \n - **Benefits:** \n \n - Reduces data transfer \n - Faster response times \n - Lower server load \n - Stays within rate limits \n \n \n #### 2 \n \n Implement Pagination with Date Ranges \n \n For large datasets, break requests into smaller chunks: \n \n `` http \n # Week 1 \n GET {{baseUrl}}/v{{version}}/orders?date=>2025-11-01&date=<2025-11-08 \n # Week 2 \n GET {{baseUrl}}/v{{version}}/orders?date=>2025-11-08&date=<2025-11-15 \n \n `` \n \n \n #### 3 \n \n Cache Static Data \n \n Cache data that changes infrequently: \n \n - Item classes (cache for 24 hours) \n - Price formulas (cache for 1 hour) \n - Customer master data (cache for 6 hours) \n \n \n #### 4 \n \n Use Specific Filters \n \n Apply filters to retrieve only the data you need: \n \n `` http \n # Instead of fetching all items and filtering locally \n GET {{baseUrl}}/v{{version}}/items \n # Fetch only what you need \n GET {{baseUrl}}/v{{version}}/items?warehouse=AALTR&quantity=>0 \n \n `` \n \n \n #### 5 \n \n Implement Exponential Backoff \n \n For transient errors (429, 503), retry with increasing delays: \n \n - 1st retry: wait 1 second \n - 2nd retry: wait 2 seconds \n - 3rd retry: wait 4 seconds \n - 4th retry: wait 8 seconds \n \n \n \n #### 6 \n \n Batch Related Requests \n \n Group related operations to minimize round trips: \n \n `` \n 1. Fetch orders: GET

```

/orders?ts=>{last_sync}\n2. For each order, fetch customer details (if
needed)\n3. Update local database in batch\n\n```\n\n#### 7\\. Monitor API
Usage\n\nTrack your API consumption:\n\n- Log request counts per endpoint\n
\n- Monitor response times\n\n- Track error rates\n\n- Set up alerts
for rate limit warnings\n\n\n---\n\n## □ Common Integration
Patterns\n\n#### Pattern 1: Initial Data Sync (First-Time
Setup)\n\n**Objective**: Load all data from Directo ERP into your
system\n\n```\nStep 1: Fetch all items\nGET /items\nStep 2: Fetch all
customers\nGET /customers\nStep 3: Fetch all stock levels\nGET
/stocklevels\nStep 4: Fetch item classes\nGET /itemclasses\nStep 5: Store
current timestamp for future incremental syncs\ntimestamp =
current_time()\n\n```\n\n**Considerations:**\n\n- May take several minutes
for large datasets\n\n- Run during off-peak hours\n\n- Implement
progress tracking\n\n- Handle timeouts gracefully\n\n\n---\n\n####
Pattern 2: Incremental Sync (Ongoing Synchronization)\n\n**Objective**: Keep
your system up-to-date with changes in Directo ERP\n\n```\nStep 1: Retrieve
last sync timestamp from your database\nlast_sync =
get_last_sync_timestamp()\nStep 2: Fetch changed items\nGET
/items?ts=>{last_sync}\nStep 3: Fetch changed customers\nGET
/customers?ts=>{last_sync}\nStep 4: Fetch changed stock levels\nGET
/stocklevels?ts=>{last_sync}\nStep 5: Fetch deleted records\nGET
/deleted?ts=>{last_sync}\nStep 6: Process changes in your system\n- Update
existing records\n- Insert new records\n- Delete removed records\nStep 7:
Update last sync timestamp\nsave_last_sync_timestamp(current_time())\n\n
```\n\n**Frequency Recommendations:**\n\n- Critical data (stock levels,
orders): Every 5-15 minutes\n\n- Master data (items, customers): Every
1-6 hours\n\n- Static data (item classes): Daily\n\n\n---\n\n####
Pattern 3: Real-Time Order Monitoring\n\n**Objective**: Monitor new orders
for immediate processing\n\n```\nStep 1: Poll orders endpoint at regular
intervals\nGET /orders?status=NEW&ts=>{last_check}\nStep 2: Process new
orders\nfor each order:\n\n- Validate order data\n\n- Check inventory
availability\n\n- Create fulfillment tasks\n\n- Update order status in
your system\nStep 3: Update last check timestamp\nlast_check =
current_time()\nStep 4: Wait for next interval (e.g., 5 minutes)\nStep 5:
Repeat from Step 1\n\n```\n\n**Best Practices:**\n\n- Poll every 5-15
minutes (not more frequently)\n\n- Use `ts` parameter to avoid
processing duplicates\n\n- Implement idempotency in your order
processing\n\n- Log all processed orders for audit trail\n\n\n---\n\n
Pattern 4: Inventory Management & Availability
Check\n\n**Objective**: Maintain accurate inventory availability in your
system\n\n```\nStep 1: Fetch current stock levels\nGET
/stocklevels?item_code={sku}\nStep 2: Fetch allocations (reserved
stock)\nGET /allocations?item_code={sku}\nStep 3: Calculate available
quantity\navailable = stock_quantity - allocated_quantity\nStep 4: Update
your inventory system\nupdate_inventory(sku, available)\nStep 5: Check for
low stock alerts\nif available < reorder_point:\n
trigger_reorder_alert(sku)\n\n```\n\n**Use Cases:**\n\n- E-commerce
available-to-promise\n\n- Warehouse management systems\n\n- Reorder
point monitoring\n\n- Multi-channel inventory sync\n\n\n---\n\n####
Pattern 5: Customer Data Synchronization (CRM Integration)\n\n**Objective**:
Keep customer data synchronized between Directo and your CRM\n\n```\nStep 1:

```

```

Fetch changed customers\nGET /customers?ts={last_sync}\nStep 2: For each
customer, sync to CRM\nfor each customer:\n if customer exists in CRM:\n
update_crm_customer(customer)\n else:\n
create_crm_customer(customer)\nStep 3: Fetch deleted customers\nGET
/deleted?resource_type=customers&ts={last_sync}\nStep 4: Remove deleted
customers from CRM\nfor each deleted_customer:\n
delete_crm_customer(deleted_customer.id)\nStep 5: Update sync
timestamp\nsave_last_sync_timestamp(current_time())\n\n```\n\n---\n\n###
Pattern 6: Invoice Export to Accounting System\n\n**Objective**: Export
invoices to external accounting software\n\n```\nStep 1: Fetch new/updated
invoices\nGET /invoices?ts=>{last_sync}\nStep 2: For each invoice, export to
accounting system\nfor each invoice:\n - Map Directo fields to accounting
system fields\n - Create invoice in accounting system\n - Create
invoice lines\n - Mark as exported in your database\nStep 3: Handle
errors\nfor each failed_invoice:\n - Log error details\n - Queue for
manual review\n - Send notification to accounting team\nStep 4: Update
sync timestamp\nsave_last_sync_timestamp(current_time())\n\n```\n\n---
\n\n### Pattern 7: Price Synchronization to E-Commerce\n\n**Objective**:
Keep product prices updated in your online store\n\n```\nStep 1: Fetch
changed items with pricing\nGET /items?ts=>{last_sync}\nStep 2: Fetch price
formulas (if using dynamic pricing)\nGET
/priceformulas?ts=>{last_sync}\nStep 3: Calculate final prices\nfor each
item:\n base_price = item.price\n if price_formula exists:\n
final_price = apply_formula(base_price, formula)\n else:\n
final_price = base_price\nStep 4: Update e-commerce platform\nfor each
item:\n update_product_price(item.code, final_price)\nStep 5: Update sync
timestamp\nsave_last_sync_timestamp(current_time())\n\n```\n\n---\n\n##
Troubleshooting Guide\n\n### Issue: Empty Response / No
Results\n\n**Possible Causes:**\n\n1. Filters too restrictive\n 2. No
data matches criteria\n 3. Incorrect date format\n 4. Filtering by
calculated field\n 5. Default filter set in ERP is too restrictive\n\n**Solutions:**\n\n```\nhttp\n# Remove filters one by one to identify the
issue\nGET /items\n# No filters\nGET
/items?warehouse=AALTR\n# Add one filter\nGET
/items?warehouse=AALTR&quantity=>0\n# Add another filter\n# Verify
date format (must be ISO 8601)\n\nWRONG: ?date=2025-11-20\n\nCORRECT:
?date=>2025-11-20T00:00:00Z\n# Check if field is calculated\n# Try filtering
by underlying stored fields instead\n\n```\n\n---\n\n### Issue:
Authentication Errors (401/403)\n\n**Possible Causes:**\n\n1. Missing API
key\n 2. Incorrect header name\n 3. Expired key\n 4.
Insufficient permissions\n\n**Solutions:**\n\n```\nhttp\n# Verify
header name (case-sensitive)\n\nWRONG: X-Api-Key: YOUR_KEY\n\nWRONG: X-
Directo-Api-Key: YOUR_KEY\n\nCORRECT: X-Directo-Key: YOUR_KEY\n# Check key
validity\n# Contact administrator to verify:\n# - Key is active\n# - Key has
permissions for the resource\n# - Key hasn't expired\n# Test with curl\ncurl
-X GET \"{{baseUrl}}/v{{version}}/items\" \\n -H \"X-Directo-Key:
YOUR_API_KEY\" \\n -H \"Accept: application/json\" \\n -v\n\n```\n\n---
\n\n### Issue: Unexpected Data Format\n\n**Possible Causes:**\n\n1. Wrong
`Accept` header\n 2. API version mismatch\n 3. Expecting different
structure\n\n**Solutions:**\n\n```\nhttp\n# Verify Accept header\nFor
JSON: Accept: application/json\nFor XML: Accept: application/xml\n# Check

```

```

API version in URL\n${baseUrl}}/v1/items (not v2 or v0)\n# Examine actual
response structure\n# Make a simple request and inspect the response\nGET
/items?code=1011\n\n``\n\n---\n\n\n### Issue: Filter Not Working as
Expected\n\n**Possible Causes:**\n\n1. Field name misspelled\n \n2. Field
is calculated (not filterable)\n \n3. Wrong operator syntax\n \n4.
Value not URL-encoded\n \n\n**Solutions:**\n\n`` http\n# Verify field
name (case-sensitive)\n\nWRONG: ?Code=1011\nCORRECT: ?code=1011\n# Check
if field exists in response\nGET /items?code=1011\n# Examine response to see
actual field names\n# Verify operator syntax\nCORRECT: ?quantity=>100
(greater than)\nCORRECT: ?quantity=<50 (less than)\nCORRECT:
?status!=CLOSED (not equal)\n# URL encode special characters\nWRONG:
?date=>2025-11-20 07:30:00\nCORRECT: ?date=>2025-11-20T07:30:00Z\n\n
``\n\n---\n\n\n### Issue: Slow Response Times\n\n**Possible Causes:**\n\n1.
Fetching too much data\n \n2. No filters applied\n \n3. Complex
queries\n \n4. Server load\n \n\n**Solutions:**\n\n`` http\n# Use
filters to reduce data volume\nSLOW: GET /items\nFASTER: GET
/items?warehouse=AALTR&closed=0\n# Use timestamp filtering for incremental
sync\nFASTER: GET /items?ts=2025-11-20T07:00:00Z\n# Break large requests
into smaller chunks\n# Instead of fetching all orders:\nGET
/orders?date=>2025-11-01&date=<2025-11-08\nGET
/orders?date=>2025-11-08&date=<2025-11-15\n# Implement caching for static
data\n# Cache item classes, price formulas for 1-24 hours\n\n``\n\n---
\n\n### Issue: Rate Limit Errors (429)\n\n**Possible Causes:**\n\n1. Too
many requests in short time\n \n2. No rate limiting in your code\n \n3. Multiple processes hitting API\n \n\n**Solutions:**\n\n`` python\n#
Implement exponential backoff\nimport time\ndef make_request_with_retry(url,
max_retries=5):\n for attempt in range(max_retries):\n response =
requests.get(url)\n if response.status_code == 200:\n
return response\n elif response.status_code == 429:\n
wait_time = 2 ** attempt # 1, 2, 4, 8, 16 seconds\n
time.sleep(wait_time)\n else:\n raise Exception(f"Error:
{response.status_code}")\n raise Exception("Max retries exceeded")\n#
Add delays between requests\ntime.sleep(1) # Wait 1 second between
requests\n# Use timestamp filtering to reduce request frequency\n# Instead
of polling every minute, poll every 5-15 minutes\n\n``\n\n---\n\n\n###
Issue: Data Inconsistency\n\n**Possible Causes:**\n\n1. Not handling
deletions\n \n2. Missing incremental updates\n \n3. Race conditions\n \n4. Not using timestamps correctly\n \n\n**Solutions:**\n\n`` http\n#
Always check for deletions\nGET /deleted?ts=>{last_sync}\n# Use consistent
timestamp handling\nlast_sync = "2025-11-20T07:00:00Z"\nGET
/items?ts=>{last_sync}\nGET /customers?ts=>{last_sync}\nGET
/deleted?ts=>{last_sync}\n# Store timestamp AFTER successful processing\n#
Not before, to avoid missing updates if processing fails\n# Implement
idempotency\n# Use unique IDs to prevent duplicate processing\n\n``\n\n---
\n\n### Issue: Missing Custom Fields\n\n**Possible Causes:**\n\n1. Custom
fields not configured in ERP\n \n2. Looking in wrong place in response\n \n3. Permissions issue\n \n\n**Solutions:**\n\n`` http\n# Custom fields
are in datafields array\nGET /items?code=1011\n# Response structure:\n{\n
 "code": "1011",\n
 "name": "Item Name",\n
 "datafields": [\n
 {\n
 "code": "CUSTOM_FIELD_1",\n
 "content":
 "Value"\n
 }\n
]\n}\n# Filter by custom fields\nGET

```

/items?CUSTOM\_FIELD\_1=Value\n# Verify custom fields exist in ERP\n# Contact administrator to confirm field names\n\n```\n\n---\n\n\n### Issue: `usagelogs` missing older entries\n\n\n\*\*Possible Cause:\*\* Log retention is capped by Administrator settings \n(\n"Log age in days"\n / \n"Log age in transactions"\n – see \n[Administratori seadistused](https://wiki.directo.ee/et/administrator\_settings)), \nnot by the API itself.\n\n\n\*\*Solution:\*\* Ask your administrator to check/raise these two settings, or \nstart pulling and archiving `/usagelogs` regularly to keep your own history.\n\n\n---\n\n\n### Support & Resources\n\n\n### Need Help?\n\n\n\*\*Technical Support:\*\*\n\n\n- Email: [info@directo.ee](https://mailto:info@directo.ee)\n\n\n- Contact your Directo account manager\n\n\n- Submit support ticket through your in Directo Chat\n\n\n\n\*\*API Issues:\*\*\n\n\n- Report bugs via your support channel\n\n\n- Include: request URL, headers, response, error message\n\n\n- Provide timestamp of the issue\n\n\n\n\*\*Feature Requests:\*\*\n\n\n- Submit through your account manager\n\n\n- Describe use case and business value\n\n\n- Include examples of desired functionality\n\n\n\n### Additional Resources\n\n\n- \*\*Directo ERP Documentation:\*\* Comprehensive guide to ERP features\n\n\n- \*\*API Changelog:\*\* Check for updates and new features\n\n\n- \*\*Integration Examples:\*\* Sample code in this collection\n\n\n- \*\*Community Forum:\*\* Share experiences with other developers (if available)\n\n\n\n### Best Practices for Support Requests\n\n\nWhen contacting support, include:\n\n\n1. \*\*Request Details:\*\*\n\n\n```\n\nMethod: GET\n\nURL: {{baseUrl}}/v1/items?warehouse=AALTR\n\nHeaders:\n\nX-Directo-Key: [REDACTED]\n\nAccept: application/json\n\n```\n\n\n2. \*\*Response Details:\*\*\n\n\nStatus Code: 400\n\n\nResponse Body: {\n"error": \n"Invalid filter"\n}\n\n\n3. \*\*Expected Behavior:\*\*\n\n\nExpected to receive items from AALTR warehouse\n\n\n4. \*\*Actual Behavior:\*\*\n\n\nReceived 400 error with \n"Invalid filter"\n message\n\n\n5. \*\*Troubleshooting Steps Taken:\*\*\n\n\n- Verified field name in response\n\n\n- Tested without filters (works)\n\n\n- Checked URL encoding\n\n\n\n---\n\n\n### Changelog & Version History\n\n\n### Version 1 (Current)\n\n\n\*\*Release Date:\*\* 2024\n\n\n\*\*Features:\*\*\n\n\n- Initial API release\n\n\n- Support for all major resources (items, customers, orders, invoices, stock levels)\n\n\n- JSON and XML response formats\n\n\n- ERP-style filtering capabilities with comparison operators\n\n\n- Timestamp-based synchronization\n\n\n- API key authentication\n\n\n- Rate limiting\n\n\n\n\*\*Endpoints:\*\*\n\n\n- `/items` - Item master data\n\n\n- `/sales\_quotations` - Sales quotations master data\n\n\n- `/customers` - Customer master data\n\n\n- `/suppliers` - Suppliers master data\n\n\n- `/contacts` - Contact persons master data\n\n\n- `/orders` - Sales orders\n\n\n- `/purchase\_orders` - Purchase orders\n\n\n- `/invoices` - Sales Invoice documents\n\n\n- `/purchase\_invoices` - Purchase Invoice documents\n\n\n- `/purchase\_payments` - Purchase Payments documents\n\n\n- `/stocklevels` - Inventory levels\n\n\n- `/stocklevels\_sn` - Inventory with serial/batch

```

numbers\n \n- `/allocations` - Inventory allocations\n \n-
`/itemclasses` - Item classifications\n \n- `/priceformulas` - Pricing
rules\n \n- `/accounts` - Financial account master data\n \n-
`/transactions` - Financial transaction documents\n \n- `/deleted` -
Deletion tracking\n \n\n**Known Limitations:**\n\n- Calculated fields
cannot be filtered\n \n- No pagination (use date ranges and timestamp
filtering)\n \n- Rate limits apply (60 requests/minute recommended)\n
\n\n---\n\n## □ Security & Compliance\n\n### Data Security\n\n-
Encryption: All API requests must use HTTPS\n \n- **Authentication**:
API key required for all requests\n \n- **Authorization**: Role-based
access control via API keys\n \n- **Audit Logging**: All API requests are
logged server-side\n \n\n### Compliance Considerations\n\n- **GDPR**:
Customer data includes personal information - handle according to GDPR
requirements\n \n- **Data Retention**: Implement appropriate data
retention policies\n \n- **Access Control**: Limit API key distribution
to authorized personnel only\n \n- **Monitoring**: Monitor API usage for
unusual patterns\n \n\n### Security Best Practices\n\n1. **Protect API
Keys:**\n \n - Never commit keys to version control\n \n -
Use environment variables or secure vaults\n \n - Rotate keys
regularly (quarterly recommended)\n \n - Use different keys for
dev/staging/production\n \n2. **Implement Access Controls:**\n \n -
Limit API key permissions to minimum required\n \n - Use
separate keys for different integrations\n \n - Revoke keys
immediately when no longer needed\n \n3. **Monitor Usage:**\n \n -
Log all API requests in your application\n \n - Set up alerts
for unusual activity\n \n - Review access logs regularly\n \n4.
Handle Data Responsibly:\n \n - Encrypt sensitive data at
rest\n \n - Implement data retention policies\n \n -
Comply with privacy regulations (GDPR, etc.)\n \n - Secure data
transmission (HTTPS only)\n \n\n---\n\n## □ Tips & Tricks\n\n### Tip
1: Discover Available Fields\n\n`` http\n# Make a request without filters
to see all available fields\nGET /items?code=1011\n# Examine the response to
discover:\n# - Field names (for filtering)\n# - Data types\n# - Custom
fields (in datafields array)\n\n``\n\n### Tip 2: Test Filters in ERP
First\n\nBefore implementing filters in your code, test them in the Directo
ERP interface. If a filter works in the ERP, it will work in the API.\n\n###
Tip 3: Use Postman Variables\n\nSet up collection variables for easy
environment switching:\n\n- `baseUrl`:\n
`https://login.directo.ee/apidirect`\n \n- `version`: `1`\n \n-
`apiKey`: `YOUR_API_KEY`\n \n\n### Tip 4: Build Filter Strings
Programmatically\n\n`` python\n# Python example\ndef
build_filter_url(base_url, filters):\n params = "&".join([f"{k}={v}"
for k, v in filters.items()])\n return f"{base_url}?{params}"\nfilters
= {\n "warehouse": "AALTR",\n "quantity": ">0",\n
"status": "ACTIVE"\n}\nurl = build_filter_url("{baseUrl}/v1/items",
filters)\n# Result:\n{{baseUrl}}/v1/items?warehouse=AALTR&quantity=>0&status=ACTIVE\n\n
``\n\n### Tip 5: Handle Timestamps Consistently\n\n`` python\n# Always use
ISO 8601 format with timezone\nfrom datetime import datetime, timezone\n#
Get current timestamp\nnow = datetime.now(timezone.utc).isoformat()\n#
Result: 2025-11-20T07:30:00+00:00\n# Or use Z notation\nnow =

```

```

datetime.now(timezone.utc).strftime("%Y-%m-%dT%H:%M:%SZ")\n# Result:
2025-11-20T07:30:00Z\n\n```\n\n### Tip 6: Implement Robust Error
Handling\n\n```python\n# Python example with comprehensive error
handling\nimport requests\nimport time\ndef fetch_directo_data(endpoint,
params=None, max_retries=3):\n url =
f"{{{baseUrl}}}/v{{{version}}}/{endpoint}"\n headers = {\n
"X-Directo-Key": "YOUR_API_KEY",\n "Accept":
"application/json"\n }\n for attempt in range(max_retries):\n
try:\n response = requests.get(url, headers=headers,
params=params, timeout=30)\n if response.status_code == 200:\n
return response.json()\n elif response.status_code == 429:\n
Rate limit - wait and retry\n wait_time = 2 ** attempt\n
time.sleep(wait_time)\n elif response.status_code in [401,
403]:\n # Auth error - don't retry\n raise
Exception(f"Authentication error: {response.status_code}")\n
else:\n # Other error - log and retry\n
print(f"Error {response.status_code}: {response.text}")\n
time.sleep(1)\n except requests.exceptions.Timeout:\n
print(f"Timeout on attempt {attempt + 1}")\n time.sleep(2)\n
except requests.exceptions.RequestException as e:\n
print(f"Request failed: {e}")\n time.sleep(2)\n raise
Exception(f"Failed after {max_retries} attempts")\n\n# Usage\nitems =
fetch_directo_data("items", params={"warehouse": "AALTR",
"quantity": ">0"})\n\n```\n\n---\n\n**Last Updated:** 29-01-2026
\n\n**API Version:** 1 \n\n**Documentation Version:** 3.0 (Enterprise
Edition)\n\n---\n\n## Quick Reference Card\n\n### Essential
Endpoints\n\n```\nItems: GET /v1/items\nCustomers: GET
/v1/customers\nOrders: GET /v1/orders\nInvoices: GET
/v1/invoices\nStock: GET /v1/stocklevels\nDeletions: GET
/v1/deleted\n\n```\n\n### Authentication\n\n```http\nX-Directo-Key:
YOUR_API_KEY\n\n```\n\n### Response Format\n\n```http\nAccept:
application/json (or application/xml)\n\n```\n\n### Common
Filters\n\n```\nExact match: ?field=value\nGreater than:
?field=>value\nLess than: ?field=Not equal:
?field=!value\nMultiple values: ?field=value1,value2\nTimestamp:
?ts=2025-11-20T07:00:00Z\n\n```\n\n### Filter Examples\n\n```http\n?warehouse=AALTR\n?quantity=>100\n?date=>2025-11-01T00:00:00&date=<2025-12-01T00:00:00\n?status=NEW,PENDING\n?country=!EE\n?ts=2025-11-20T07:00:00Z\n\n```\n\n### Remember\n\nAll fields can be filtered (except calculated fields) \n\nFilters mirror ERP system behavior \n\nUse timestamp filtering for sync \n\nURL encode special characters \n\nImplement rate limiting (60 req/min) \n\nHandle errors with exponential backoff\n\n---\n\nThis
documentation is maintained by the Directo development team. For questions
or feedback, contact_
[<i>info@directo.ee</i>](https://mailto:info@directo.ee)",
 "schema":
"https://schema.getpostman.com/json/collection/v2.1.0/collection.json",
 "_exporter_id": "14246513"
},
"item": [
{

```

```
"name": "apidirect",
"item": [
 {
 "name": "v{version}",
 "item": [
 {
 "name": "sales_quotations",
 "item": [
 {
 "name": "All sales quotations",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/sales_quotations",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "sales_quotations"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "row_item",
 "value": "9810",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
 "disabled": true
 },
 {
 "key": "row_description",
 "value": "Maja",
```

```
 "disabled": true
 },
 {
 "key": "stock",
 "value": "AALTR,AIPM",
 "disabled": true
 },
 {
 "key": "row_quantity",
 "value": "as",
 "disabled": true
 },
 {
 "key": "country",
 "value": "!EE",
 "disabled": true
 },
 {
 "key": "date",
 "value": "<2025-09-03T00:00:00",
 "disabled": true
 },
 {
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
 },
 {
 "key": "row_stock",
 "value": "223TEST",
 "disabled": true
 },
 {
 "key": "probability",
 "value": ">50",
 "type": "text",
 "disabled": true
 }
]
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },

```

```

 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/sales_quotations?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "sales_quotations"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 }
},
"status": "OK",
"code": 200,
"_postman_previewlanguage": "xml",
"header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
"cookie": [],
"body": ""
}
]
}
},
{
 "name": "orders",
 "item": [
 {
 "name": "All orders",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",

```

```
 "value": "application/xml"
 }
],
"url": {
 "raw": "{{baseUrl}}/v{{version}}/orders",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "orders"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "19900274",
 "disabled": true
 },
 {
 "key": "row_item",
 "value": "9810",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
 "disabled": true
 },
 {
 "key": "row_description",
 "value": "Maja",
 "disabled": true
 },
 {
 "key": "stock",
 "value": "AALTR,AIPM",
 "disabled": true
 },
 {
 "key": "row_quantity",
 "value": "as",
 "disabled": true
 },
 {
 "key": "country",
 "value": "!EE",
```

```

 "disabled": true
 },
 {
 "key": "date",
 "value": "<2025-09-03T00:00:00",
 "disabled": true
 },
 {
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
 },
 {
 "key": "row_stock",
 "value": "223TEST",
 "disabled": true
 },
 {
 "key": "row_r_discount",
 "value": ">0",
 "type": "text",
 "disabled": true
 },
 {
 "key": "order_type",
 "value": null,
 "type": "text",
 "disabled": true
 }
]
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
]
 },
 "url": {
 "raw":

```

```

"{{baseUrl}}/v{{version}}/orders?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "orders"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
},
"status": "OK",
"code": 200,
"_postman_previewlanguage": "xml",
"header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
"cookie": [],
"body": ""
}
]
}
},
{
 "name": "invoices",
 "item": [
 {
 "name": "All invoices",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/invoices",
 "host": [
 "{{baseUrl}}"
],
 "path": [

```

```
 "v{{version}}",
 "invoices"
],
 "query": [
 {
 "key": "ts",
 "value": ">2025-01-01T08:11:05.129Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "row_item",
 "value": "9810",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
 "disabled": true
 },
 {
 "key": "row_description",
 "value": "Maja",
 "disabled": true
 },
 {
 "key": "date",
 "value": "2025-09-02",
 "disabled": true
 },
 {
 "key": "stock",
 "value": "AALTR,AIPM",
 "disabled": true
 },
 {
 "key": "row_quantity",
 "value": ">100",
 "disabled": true
 },
 {
 "key": "country",
 "value": "!EE",
 "disabled": true
 },
 {
 "key": "row_item",
```

```

 "value": "000033AAAXX",
 "disabled": true
 }
]
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/invoices?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "invoices"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 }
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],

```

```
 "body": ""
 }
]
 },
 {
 "name": "purchase_orders",
 "item": [
 {
 "name": "All purchase orders",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/purchase_orders",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "purchase_orders"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "row_item",
 "value": "9810",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
 "disabled": true
 },
 {
 "key": "row_description",
```

```
 "value": "Maja",
 "disabled": true
 },
 {
 "key": "stock",
 "value": "AALTR,AIPM",
 "disabled": true
 },
 {
 "key": "row_quantity",
 "value": "as",
 "disabled": true
 },
 {
 "key": "country",
 "value": "!EE",
 "disabled": true
 },
 {
 "key": "date",
 "value": "<2025-09-03T00:00:00",
 "disabled": true
 },
 {
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
 },
 {
 "key": "row_stock",
 "value": "223TEST",
 "disabled": true
 }
]
}
},
"response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security"
 }
]
 }
 }
]
```

scheme: apikey"

```

 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/purchase_orders?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "purchase_orders"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 }
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
 }
]
},
{
 "name": "stock_receipts",
 "item": [
 {
 "name": "All stock receipts",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
]
 },
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/stock_receipts",

```

```
"host": [
 "{{baseUrl}}"
],
"path": [
 "v{{version}}",
 "stock_receipts"
],
"query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "row_item",
 "value": "9810",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
 "disabled": true
 },
 {
 "key": "row_description",
 "value": "Maja",
 "disabled": true
 },
 {
 "key": "stock",
 "value": "AALTR,AIPM",
 "disabled": true
 },
 {
 "key": "row_quantity",
 "value": "as",
 "disabled": true
 },
 {
 "key": "country",
 "value": "!EE",
 "disabled": true
 },
 {
 "key": "date",
 "value": "<2025-09-03T00:00:00",
```

```

 "disabled": true
 },
 {
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
 },
 {
 "key": "row_stock",
 "value": "223TEST",
 "disabled": true
 }
]
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
]
 },
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/stock_receipts",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "stock_receipts"
]
 }
 },
 {
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
]
 }
]
}

```

```
],
 "cookie": [],
 "body": ""
 }
]
},
{
 "name": "purchase_invoices",
 "item": [
 {
 "name": "All purchase invoices",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/purchase_invoices",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "purchase_invoices"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "row_item",
 "value": "9810",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
 "disabled": true
 }
]
 }
 }
 }
]
}
```

```
{
 "key": "row_description",
 "value": "Maja",
 "disabled": true
},
{
 "key": "stock",
 "value": "AALTR,AIPM",
 "disabled": true
},
{
 "key": "row_quantity",
 "value": "as",
 "disabled": true
},
{
 "key": "country",
 "value": "!EE",
 "disabled": true
},
{
 "key": "date",
 "value": "<2025-09-03T00:00:00",
 "disabled": true
},
{
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
},
{
 "key": "row_stock",
 "value": "223TEST",
 "disabled": true
},
{
 "key": "row_invoicenum",
 "value": "500226",
 "type": "text",
 "disabled": true
}
]
},
"response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
```

```

 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
"url": {
 "raw":
"{{baseUrl}}/v{{version}}/purchase_invoices?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "purchase_invoices"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
}
},
"status": "OK",
"code": 200,
"_postman_previewlanguage": "xml",
"header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
"cookie": [],
"body": ""
}
]
}
},
{
 "name": "purchase_payments",
 "item": [
 {
 "name": "All purchase payments",
 "request": {
 "method": "GET",

```

```
"header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
"url": {
 "raw": "{{baseUrl}}/v{{version}}/purchase_payments",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "purchase_payments"
],
 "query": [
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "confirmed",
 "value": "UUS",
 "disabled": true
 },
 {
 "key": "date",
 "value": "<2025-09-03T00:00:00",
 "disabled": true
 },
 {
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
 },
 {
 "key": "object",
 "value": "!EE",
 "disabled": true
 },
 {
 "key": "project",
 "value": "AALTR,AIPM",
 "disabled": true
 },
],
}
```

```

 {
 "key": "row_rn",
 "value": "Maja",
 "disabled": true
 },
 {
 "key": "row_suppliercode",
 "value": "9810",
 "disabled": true
 },
 {
 "key": "row_quantity",
 "value": "as",
 "disabled": true
 },
 {
 "key": "row_object",
 "value": "223TEST",
 "disabled": true
 }
]
}
},
"response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/purchase_payments?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "purchase_payments"
],
 "query": [

```

```

 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
}
},
"status": "OK",
"code": 200,
"_postman_previewlanguage": "xml",
"header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
"cookie": [],
"body": ""
}
]
}
},
{
 "name": "customers",
 "item": [
 {
 "name": "All customers",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/customers",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "customers"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 },

```

```

 {
 "key": "code",
 "value": null,
 "disabled": true
 },
 {
 "key": "closed",
 "value": "0",
 "disabled": true
 }
]
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/customers?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "customers"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 }
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",

```

```
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
 }
]
 },
 {
 "name": "suppliers",
 "item": [
 {
 "name": "All suppliers",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/suppliers",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "suppliers"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 },
 {
 "key": "data_row_code",
 "value": "PROMO,HANKIJA_LINK_SIS",
 "disabled": true
 },
 {
 "key": "closed",
 "value": "0",
 "disabled": true
 }
]
 }
 }
 }
]
 }
]
```

```

]
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/suppliers?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "suppliers"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 }
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
 }
]
}

```

```
 }
]
},
{
 "name": "contacts",
 "item": [
 {
 "name": "All contacts",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/contacts",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "contacts"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 },
 {
 "key": "code",
 "value": null,
 "disabled": true
 },
 {
 "key": "closed",
 "value": "0",
 "disabled": true
 }
]
 }
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
```

```

 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
"url": {
 "raw":
"{{baseUrl}}/v{{version}}/contacts?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "contacts"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
}
},
"status": "OK",
"code": 200,
"_postman_previewlanguage": "xml",
"header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
"cookie": [],
"body": ""
}
]
}
},
{
 "name": "items",
 "item": [
 {
 "name": "All items",
 "event": [
 {

```

```
 "listen": "test",
 "script": {
 "exec": [
 ""
],
 "type": "text/javascript",
 "packages": {},
 "requests": {}
 }
 },
 {
 "listen": "prerequisite",
 "script": {
 "exec": [
 ""
],
 "type": "text/javascript",
 "packages": {},
 "requests": {}
 }
 }
],
"request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "",
 "value": "",
 "type": "text",
 "disabled": true
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/items",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "items"
],
 "query": [
 {
 "key": "code",
 "value": "",
 "disabled": true
 },

```

```
{
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 },
 {
 "key": "stock",
 "value": "AALTR,AIPM",
 "disabled": true
 },
 {
 "key": "country",
 "value": "EE",
 "disabled": true
 },
 {
 "key": "class",
 "value": "AUTOD",
 "disabled": true
 },
 {
 "key": "closed",
 "value": "1",
 "disabled": true
 },
 {
 "key": "name",
 "value": "%name%",
 "type": "text",
 "disabled": true
 },
 {
 "key": "description",
 "value": "%description%",
 "disabled": true
 }
]
},
"response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
```

```

 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/items?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "items"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 }
},
"status": "OK",
"code": 200,
"_postman_previewlanguage": "xml",
"header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
"cookie": [],
"body": ""
}
]
}
},
{
 "name": "itemclasses",
 "item": [
 {
 "name": "All itemclasses",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
]
 }
 }
]
}

```

```

],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/itemclasses",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "itemclasses"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 },
 {
 "key": "masters",
 "value": "CON",
 "disabled": true
 },
 {
 "key": "code",
 "value": null,
 "disabled": true
 }
]
 }
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
]
 },
 "url": {
 "raw":
 "{{baseUrl}}/v{{version}}/itemclasses?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"

```

```

],
 "path": [
 "v{{version}}",
 "itemclasses"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
}
]
}
],
{
 "name": "priceformulas",
 "item": [
 {
 "name": "All priceformulas",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/priceformulas",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "priceformulas"
],

```

```

 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 },
 {
 "key": "row_item",
 "value": "0009",
 "disabled": true
 }
]
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/priceformulas?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "priceformulas"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 }
 },
 "status": "OK",
 "code": 200,

```

```
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
 }
}
]
},
{
 "name": "stocklevels",
 "item": [
 {
 "name": "All stocklevels",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/stocklevels",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "stocklevels"
],
 "query": [
 {
 "key": "code",
 "value": "_4008110308067",
 "disabled": true
 },
 {
 "key": "stock",
 "value": "123TEST",
 "disabled": true
 }
]
 }
 }
 }
]
},
"response": [
```

```

{
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/stocklevels",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "stocklevels"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z",
 "disabled": true
 }
]
 }
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
}
],
{
 "name": "All stocklevels with serialnumbers/batchnumbers",
 "request": {

```

```
"method": "GET",
"header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
"url": {
 "raw": "{{baseUrl}}/v{{version}}/stocklevels_sn",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "stocklevels_sn"
],
 "query": [
 {
 "key": "code",
 "value": "",
 "disabled": true
 },
 {
 "key": "sn",
 "value": "E101-10",
 "disabled": true
 },
 {
 "key": "stock",
 "value": "TL1,TL",
 "disabled": true
 }
]
},
"response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security"
 }
]
 }
 }
]
scheme: apikey"
```

```
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/stocklevels_sn",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "stocklevels_sn"
]
 }
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
}
]
},
{
 "name": "All Allocations",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/allocations",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "allocations"
]
 },
 "query": [
 {
 "key": "code",
 "value": null,
 "disabled": true
 }
]
 }
}
```

```

]
 }
 },
 "response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security
scheme: apikey"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/allocations",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "allocations"
]
 }
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
 }
]
 }
},
{
 "name": "accounts",
 "item": [
 {

```

```
"name": "All accounts",
"request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/accounts",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "accounts"
],
 "query": [
 {
 "key": "code",
 "value": "",
 "disabled": true
 },
 {
 "key": "class",
 "value": "",
 "disabled": true
 },
 {
 "key": "closed",
 "value": "1",
 "disabled": true
 }
]
 }
},
"response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security"
```

```

scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/accounts?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "accounts"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
}
]
}
},
{
 "name": "transactions",
 "item": [
 {
 "name": "All transactions",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
]
 },
 "url": {

```

```
"raw": "{{baseUrl}}/v{{version}}/transactions",
"host": [
 "{{baseUrl}}"
],
"path": [
 "v{{version}}",
 "transactions"
],
"query": [
 {
 "key": "number",
 "value": "1011",
 "disabled": true
 },
 {
 "key": "type",
 "value": "FIN,LAEK",
 "disabled": true
 },
 {
 "key": "date",
 "value": "<2023-05-10T09:13:57",
 "disabled": true
 },
 {
 "key": "row_account",
 "value": "111281",
 "disabled": true
 },
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 }
]
},
"response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security"
```

```

scheme: apikey"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/transactions?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "transactions"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
}
]
},
{
 "name": "deletions",
 "item": [
 {
 "name": "All deleted",
 "request": {
 "method": "GET",
 "header": [],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/deleted",
 "host": [
 "{{baseUrl}}"
],
 "path": [

```

```
 "v{{version}}",
 "deleted"
],
 "query": [
 {
 "key": "document",
 "value": "maksuvalem,inventar",
 "disabled": true
 },
 {
 "key": "user",
 "value": "!SUPER",
 "disabled": true
 },
 {
 "key": "ts",
 "value": ">2020-09-23T16:37:09",
 "disabled": true
 },
 {
 "key": "code",
 "value": null,
 "disabled": true
 },
 {
 "key": "number",
 "value": null,
 "disabled": true
 }
]
},
"response": [
 {
 "name": "OK",
 "originalRequest": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 },
 {
 "key": "X-Directo-Key",
 "value": "<API Key>",
 "description": "Added as a part of security"
 }
]
 },
 "raw": ""
 }
]
```

scheme: apikey"

```

 "{{baseUrl}}/v{{version}}/deleted?ts=1949-07-30T08:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "deleted"
],
 "query": [
 {
 "key": "ts",
 "value": "1949-07-30T08:11:05.129Z"
 }
]
 },
 "status": "OK",
 "code": 200,
 "_postman_previewlanguage": "xml",
 "header": [
 {
 "key": "Content-Type",
 "value": "application/xml"
 }
],
 "cookie": [],
 "body": ""
}
]
},
{
 "name": "financial_budgets",
 "item": [
 {
 "name": "All financial budgets",
 "request": {
 "method": "GET",
 "header": [],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/financial_budgets",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "financial_budgets"
]
 }
 }
 }
]
},

```

```
 "response": []
 }
]
 },
 {
 "name": "events",
 "item": [
 {
 "name": "All events",
 "request": {
 "method": "GET",
 "header": [],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/events",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "events"
]
 }
 },
 "response": []
 }
]
 },
 {
 "name": "receipts",
 "item": [
 {
 "name": "All receipts",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/receipts",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "receipts"
],
 "query": [
 {

```

```

 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 },
 {
 "key": "code",
 "value": null,
 "disabled": true
 },
 {
 "key": "closed",
 "value": "0",
 "disabled": true
 }
]
 },
 "response": []
}
]
},
{
 "name": "deliveries",
 "item": [
 {
 "name": "All deliveries",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/deliveries",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "deliveries"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05.129Z",
 "disabled": true
 },
 {
 "key": "code",

```

```
 "value": "",
 "disabled": true
 },
 {
 "key": "confirmed",
 "value": "0",
 "disabled": true
 }
]
},
"response": []
}
]
},
{
 "name": "movements",
 "item": [
 {
 "name": "All movements",
 "request": {
 "method": "GET",
 "header": [],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/movements",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "movements"
]
 }
 },
 "response": []
 }
]
},
{
 "name": "objects",
 "item": [
 {
 "name": "All objects",
 "request": {
 "method": "GET",
 "header": [],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/objects",
 "host": [
 "{{baseUrl}}"
],

```

```
 "path": [
 "v{{version}}",
 "objects"
]
 },
 "response": []
}
]
},
{
 "name": "projects",
 "item": [
 {
 "name": "All projects",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/projects",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "projects"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "row_item",
 "value": "9810",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
```

```
 "disabled": true
 },
 {
 "key": "row_description",
 "value": "Maja",
 "disabled": true
 },
 {
 "key": "stock",
 "value": "AALTR,AIPM",
 "disabled": true
 },
 {
 "key": "row_quantity",
 "value": "as",
 "disabled": true
 },
 {
 "key": "country",
 "value": "!EE",
 "disabled": true
 },
 {
 "key": "date",
 "value": "<2025-09-03T00:00:00",
 "disabled": true
 },
 {
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
 },
 {
 "key": "row_stock",
 "value": "223TEST",
 "disabled": true
 }
]
 },
 "response": []
}
]
},
{
 "name": "resources",
 "item": [
 {
 "name": "All resources",
 "request": {
 "method": "GET",
```

```
"header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
"url": {
 "raw": "{{baseUrl}}/v{{version}}/resources",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "resources"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
 "disabled": true
 },
 {
 "key": "date",
 "value": "<2025-09-03T00:00:00",
 "disabled": true
 },
 {
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
 },
 {
 "key": "confirmed",
 "value": "1",
 "type": "text",
 "disabled": true
 },
 {
 "key": "row_acceptedhours",
 "value": ">0",
 "type": "text",
```

```
 "disabled": true
 },
 {
 "key": "row_description",
 "value": "Maja",
 "disabled": true
 }
]
 },
 "response": []
}
],
},
{
 "name": "sales_contracts",
 "item": [
 {
 "name": "All sales contracts",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/sales_contracts",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "sales_contracts"
],
 "query": [
 {
 "key": "ts",
 "value": "2025-01-01T08:11:05Z",
 "disabled": true
 },
 {
 "key": "number",
 "value": "!1908694",
 "disabled": true
 },
 {
 "key": "status",
 "value": "UUS",
 "disabled": true
 }
]
 }
 }
 }
]
}
```

```

 },
 {
 "key": "row_description",
 "value": "Maja",
 "disabled": true
 },
 {
 "key": "date",
 "value": "<2025-09-03T00:00:00",
 "disabled": true
 },
 {
 "key": "date",
 "value": ">2025-09-01T00:00:00",
 "disabled": true
 }
]
}
},
"response": []
}
]
},
{
 "name": "replacementlogs",
 "item": [
 {
 "name": "All replacementlogs",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/replacementlogs",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "replacementlogs"
],
 "query": [
 {
 "key": "nr",
 "value": null,
 "type": "text",
 "disabled": true
 }
]
 }
 }
 }
]
}

```

```
 },
 {
 "key": "document",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "ts",
 "value": null,
 "type": "text",
 "disabled": true
 }
]
},
"response": []
}
]
},
{
 "name": "users",
 "item": [
 {
 "name": "All users",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/users",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "users"
],
 "query": [
 {
 "key": "ts",
 "value": "",
 "type": "text",
 "disabled": true
 },
 {
 "key": "code",
```

```

 "value": "xx",
 "type": "text",
 "disabled": true
 },
 {
 "key": "employee",
 "value": "!1",
 "type": "text",
 "disabled": true
 }
]
}
},
"response": []
}
]
},
{
 "name": "recipes",
 "item": [
 {
 "name": "All recipes",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/recipes",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "recipes"
],
 "query": [
 {
 "key": "ts",
 "value": "",
 "type": "text",
 "disabled": true
 },
 {
 "key": "code",
 "value": "00001",
 "type": "text",
 "disabled": true
 }
]
 }
 }
 }
]
}

```

```
 }
]
},
{
 "name": "nettings",
 "item": [
 {
 "name": "All nettings",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/nettings",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "nettings"
],
 "query": [
 {
 "key": "ts",
 "value": "",
 "type": "text",
 "disabled": true
 },
 {
 "key": "number",
 "value": "1,29",
 "type": "text",
 "disabled": true
 },
 {
 "key": "row_object",
 "value": "ISIK",
 "type": "text",
 "disabled": true
 },
 {
 "key": "row_purchaseinvoice",
```

```

 "value": "201400035",
 "type": "text",
 "disabled": true
 },
 {
 "key": "suppliercode",
 "value": "00009",
 "type": "text",
 "disabled": true
 }
]
}
},
"response": []
}
]
},
{
 "name": "usagelogs",
 "item": [
 {
 "name": "All usagelogs",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw":
"{{baseUrl}}/v{{version}}/usagelogs?requeststart=>2026-06-01T09:11:05.129Z",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "usagelogs"
],
 "query": [
 {
 "key": "requeststart",
 "value": ">2026-06-01T09:11:05.129Z",
 "type": "text"
 },
 {
 "key": "requestend",
 "value": null,
 "type": "text",
 "disabled": true
 }
]
 }
 }
 }
]
}

```

```
 },
 {
 "key": "nr",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "name",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "ip",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "cu",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "datafield1",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "datafield2",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "code",
 "value": null,
 "type": "text",
 "disabled": true
 }
]
},
"response": []
}
]
```

```
"name": "expenses",
"item": [
 {
 "name": "All expenses",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/expenses",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "expenses"
],
 "query": [
 {
 "key": "number",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "expenders",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "customercode",
 "value": "",
 "type": "text",
 "disabled": true
 },
 {
 "key": "customername",
 "value": "",
 "type": "text",
 "disabled": true
 },
 {
 "key": "date",
 "value": null,
 "type": "text",
 "disabled": true
 }
]
 }
 }
 }
]
```

```
 },
 {
 "key": "type",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "object",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "status",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "cu",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "ts",
 "value": null,
 "type": "text",
 "disabled": true
 },
 {
 "key": "row_account",
 "value": "7052",
 "type": "text",
 "disabled": true
 },
 {
 "key": "row_suppliercode",
 "value": "00",
 "type": "text",
 "disabled": true
 }
]
},
"response": []
}
]
```

```
"name": "approvals",
"item": [
 {
 "name": "All approvals",
 "request": {
 "method": "GET",
 "header": [
 {
 "key": "Accept",
 "value": "application/xml"
 }
],
 "url": {
 "raw": "{{baseUrl}}/v{{version}}/approvals",
 "host": [
 "{{baseUrl}}"
],
 "path": [
 "v{{version}}",
 "approvals"
],
 "query": [
 {
 "key": "module",
 "value": "",
 "disabled": true
 },
 {
 "key": "number",
 "value": "",
 "type": "text",
 "disabled": true
 },
 {
 "key": "id",
 "value": "",
 "type": "text",
 "disabled": true
 },
 {
 "key": "employee",
 "value": "",
 "type": "text",
 "disabled": true
 },
 {
 "key": "role",
 "value": "digiallkiri",
 "type": "text",
 "disabled": true
 }
]
 }
 }
 }
]
```

```
{
 "key": "signaturestatus",
 "value": "",
 "type": "text",
 "disabled": true
},
{
 "key": "signature",
 "value": "",
 "type": "text",
 "disabled": true
},
{
 "key": "signaturetime",
 "value": "",
 "type": "text",
 "disabled": true
},
{
 "key": "ts",
 "value": null,
 "type": "text",
 "disabled": true
},
{
 "key": "changedby",
 "value": "",
 "type": "text",
 "disabled": true
},
{
 "key": "externalid",
 "value": null,
 "type": "text",
 "disabled": true
}
]
},
"response": []
}
]
}
]
}
]
}
],
"auth": {
 "type": "apikey",
 "apikey": [
```

```

 {
 "key": "value",
 "value": "{{APIKey}}",
 "type": "string"
 },
 {
 "key": "key",
 "value": "X-Directo-Key",
 "type": "string"
 },
 {
 "key": "in",
 "value": "header",
 "type": "string"
 }
]
},
"event": [
 {
 "listen": "prerequisite",
 "script": {
 "type": "text/javascript",
 "packages": {},
 "requests": {},
 "exec": [
 ""
]
 }
 },
 {
 "listen": "test",
 "script": {
 "type": "text/javascript",
 "requests": {},
 "exec": [
 "// =====",
 "// DIRECTO API - COMPREHENSIVE TEST SUITE",
 "// =====",
 "// These tests run for all requests in the collection",
 "// Covers: Status codes, response structure, data validation,
error handling",
 "",
 "// Test 1: Response Status Code Validation",
 "pm.test(\"Status code is 200 OK\", function () {",
 " pm.response.to.have.status(200);",
 "});",
 "",
 "// Test 2: Response Time Performance Check",
 "pm.test(\"Response time is acceptable (under 5 seconds)\",
function () {",
 " pm.expect(pm.response.responseTime).to.be.below(5000);",

```

```

 "});",
 "",
 "// Test 3: Content-Type Header Validation",
 "pm.test(\"Content-Type header is present\", function () {",
 " pm.response.to.have.header(\"Content-Type\");",
 "});",
 "",
 "// Test 4: Response Body Exists",
 "pm.test(\"Response body is not empty\", function () {",
 " pm.expect(pm.response.text()).to.not.be.empty;",
 "});",
 "",
 "// Test 5: Valid JSON or XML Response",
 "pm.test(\"Response is valid JSON or XML format\", function () {",
 " const contentType = pm.response.headers.get(\"Content-
Type\");",
 " ",
 " if (contentType &&
contentType.includes(\"application/json\")) {",
 " pm.expect(() => pm.response.json()).to.not.throw();",
 " } else if (contentType &&
(contentType.includes(\"application/xml\") ||
contentType.includes(\"text/xml\"))) {",
 " pm.expect(() =>
xml2Json(pm.response.text())).to.not.throw();",
 " }",
 "});",
 "",
 "// =====",
 "// RESPONSE STRUCTURE & DATA VALIDATION",
 "// =====",
 "",
 "try {",
 " const contentType = pm.response.headers.get(\"Content-
Type\");",
 " let responseData;",
 " let isJson = false;",
 " ",
 " // Parse response based on content type",
 " if (contentType &&
contentType.includes(\"application/json\")) {",
 " responseData = pm.response.json();",
 " isJson = true;",
 " ",
 " // Test 6: JSON Response Structure",
 " pm.test(\"JSON response has valid structure\", function
() {",
 " ",
 " pm.expect(responseData).to.be.an('object').or.to.be.an('array');",
 " ",
 " });",
 " ",
 " ",

```

```

 " } else if (contentType &&
(contentType.includes(\"application/xml\") ||
contentType.includes(\"text/xml\"))) {\",
 \" responseData = xml2Json(pm.response.text());\",
 \" \",
 \" // Test 6: XML Response Structure\",
 \" pm.test(\"XML response has valid structure\", function ()
{\",
 \" pm.expect(responseData).to.be.an('object');\",
 \" });\",
 \" }\",
 \" \",
 \" // =====\",
 \" // ENDPOINT-SPECIFIC DATA VALIDATION\",
 \" // =====\",
 \" \",
 \" const requestName = pm.info.requestName.toLowerCase();\",
 \" \",
 \" // Test 7: Response Data Type and Array Validation\",
 \" if (responseData) {\",
 \" if (Array.isArray(responseData)) {\",
 \" pm.test(\"Response is an array with items\", function
() {\",
 \" pm.expect(responseData).to.be.an('array');\",
 \" });\",
 \" \",
 \" if (responseData.length > 0) {\",
 \" pm.test(\"Array contains valid objects\",
function () {\",
 \" pm.expect(responseData[0]).to.be.an('object');\",
 \" });\",
 \" \",
 \" // Test 8: Data Integrity - Check for required
fields based on endpoint\",
 \" if (requestName.includes(\"item\")) {\",
 \" pm.test(\"Items have required fields\",
function () {\",
 \" const item = responseData[0];\",
 \" pm.expect(item).to.have.property('id').or.to.have.property('itemId').or.to.h
ave.property('code');\",
 \" });\",
 \" \",
 \" // Type checking for items\",
 \" pm.test(\"Item fields have correct data
types\", function () {\",
 \" const item = responseData[0];\",
 \" if (item.id)
pm.expect(item.id).to.satisfy(val => typeof val === 'string' || typeof val
=== 'number');\",

```

```

 " if (item.name)
pm.expect(item.name).to.be.a('string');"
 " if (item.price)
pm.expect(item.price).to.be.a('number').or.to.be.a('string');"
 " });";
 " }";
 " ",
 " if (requestName.includes(\"customer\")) {",
 " pm.test(\"Customers have required fields\",
function () {",
 " const customer = responseData[0];",
 " pm.expect(customer).to.have.property('id').or.to.have.property('customerId')
.or.to.have.property('code');"
 " });";
 " ",
 " pm.test(\"Customer fields have correct data
types\", function () {",
 " const customer = responseData[0];",
 " if (customer.id)
pm.expect(customer.id).to.satisfy(val => typeof val === 'string' || typeof
val === 'number');"
 " if (customer.name)
pm.expect(customer.name).to.be.a('string');"
 " if (customer.email)
pm.expect(customer.email).to.be.a('string');"
 " });";
 " }";
 " ",
 " if (requestName.includes(\"order\")) {",
 " pm.test(\"Orders have required fields\",
function () {",
 " const order = responseData[0];",
 " pm.expect(order).to.have.property('id').or.to.have.property('orderId').or.to
.have.property('orderNumber');"
 " });";
 " ",
 " pm.test(\"Order fields have correct data
types\", function () {",
 " const order = responseData[0];",
 " if (order.id)
pm.expect(order.id).to.satisfy(val => typeof val === 'string' || typeof val
=== 'number');"
 " if (order.date)
pm.expect(order.date).to.be.a('string');"
 " if (order.total)
pm.expect(order.total).to.be.a('number').or.to.be.a('string');"
 " });";
 " }";
 " ",
 " ",

```

```

 " if (requestName.includes(\"invoice\")) {",
 " pm.test(\"Invoices have required fields\",
function () {",
 " const invoice = responseData[0];",
 "
pm.expect(invoice).to.have.property('id').or.to.have.property('invoiceId').o
r.to.have.property('invoiceNumber');",
 " });",
 " ",
 " pm.test(\"Invoice fields have correct data
types\", function () {",
 " const invoice = responseData[0];",
 " if (invoice.id)
pm.expect(invoice.id).to.satisfy(val => typeof val === 'string' || typeof
val === 'number');",
 " if (invoice.date)
pm.expect(invoice.date).to.be.a('string');",
 " if (invoice.amount)
pm.expect(invoice.amount).to.be.a('number').or.to.be.a('string');",
 " });",
 " }",
 " ",
 " if (requestName.includes(\"stocklevel\")) {",
 " pm.test(\"Stock levels have required
fields\", function () {",
 " const stock = responseData[0];",
 "
pm.expect(stock).to.have.property('itemId').or.to.have.property('item').or.t
o.have.property('quantity');",
 " });",
 " ",
 " pm.test(\"Stock level fields have correct
data types\", function () {",
 " const stock = responseData[0];",
 " if (stock.quantity)
pm.expect(stock.quantity).to.be.a('number').or.to.be.a('string');",
 " if (stock.itemId)
pm.expect(stock.itemId).to.satisfy(val => typeof val === 'string' || typeof
val === 'number');",
 " });",
 " }",
 " ",
 " if (requestName.includes(\"allocation\")) {",
 " pm.test(\"Allocations have required fields\",
function () {",
 " const allocation = responseData[0];",
 "
pm.expect(allocation).to.have.property('id').or.to.have.property('itemId').o
r.to.have.property('quantity');",
 " });",
 " }",

```

```

"
"
"
fields\"", function () {",
"
"
const itemClass = responseData[0];",
"
pm.expect(itemClass).to.have.property('id').or.to.have.property('code').or.to
o.have.property('name');",
"
"
});",
"
"
"
"
"
if (requestName.includes(\"priceformula\")) {",
"
pm.test(\"Price formulas have required
fields\"", function () {",
"
"
const formula = responseData[0];",
"
pm.expect(formula).to.have.property('id').or.to.have.property('code').or.to.
have.property('formula');",
"
"
});",
"
"
"
"
"
if (requestName.includes(\"deleted\")) {",
"
pm.test(\"Deleted records have required
fields\"", function () {",
"
"
const deleted = responseData[0];",
"
pm.expect(deleted).to.have.property('id').or.to.have.property('deletedId').o
r.to.have.property('type');",
"
"
});",
"
"
"
"
"
} else {",
"
pm.test(\"Empty array response is valid (no data
available)\", function () {",
"
pm.expect(responseData).to.be.an('array').that.is.empty;",
"
"
});",
"
"
"
"
} else if (typeof responseData === 'object') {",
"
pm.test(\"Response is a valid object\", function ()
{",
"
"
pm.expect(responseData).to.be.an('object');",
"
"
});",
"
"
"
"
// Check if it's an error response",
"
if (responseData.error || responseData.message) {",
"
pm.test(\"Error response contains error
details\"", function () {",
"
pm.expect(responseData).to.have.property('error').or.to.have.property('messa
ge');",
"
"
"
});",

```

```

 " }",
 " }",
 " }",
 " ",
 " // =====",
 " // DATA INTEGRITY CHECKS",
 " // =====",
 " ",
 " // Test 9: No Null or Undefined Critical Values",
 " if (Array.isArray(responseData) && responseData.length > 0)
{",
 " pm.test(\"Critical fields are not null or undefined\",
function () {",
 " responseData.forEach((item, index) => {",
 " if (item.id !== undefined) {",
 " pm.expect(item.id, `Item ${index} has
null/undefined id`).to.not.be.null;",
 " pm.expect(item.id, `Item ${index} has
null/undefined id`).to.not.be.undefined;",
 " }",
 " });",
 " });",
 " }",
 " ",
 " // Test 10: Consistent Data Structure Across Array Items",
 " if (Array.isArray(responseData) && responseData.length > 1)
{",
 " pm.test(\"Array items have consistent structure\",
function () {",
 " const firstItemKeys =
Object.keys(responseData[0]).sort();",
 " const allConsistent = responseData.every(item => {",
 " const itemKeys = Object.keys(item).sort();",
 " return JSON.stringify(firstItemKeys) ===
JSON.stringify(itemKeys);",
 " });",
 " ",
 " // This is a soft check - log warning if inconsistent
but don't fail",
 " if (!allConsistent) {",
 " console.warn(\"Warning: Array items have
inconsistent structure\");",
 " }",
 " pm.expect(true).to.be.true; // Always pass but log
the warning",
 " });",
 " }",
 " ",
 " } catch (error) {",
 " // =====",
 " // ERROR HANDLING",

```

```

" // =====",
" ",
" pm.test("Response parsing error handled gracefully",
function () {",
" console.error("Error parsing response:",
error.message);",
" console.error("Response body:",
pm.response.text().substring(0, 500));",
" ",
" // Check if it's an authentication error",
" if (pm.response.code === 401 || pm.response.code === 403)
{",
" pm.expect.fail("Authentication failed - check X-
Directo-Key header");",
" }",
" ",
" // Check if it's a server error",
" if (pm.response.code >= 500) {",
" pm.expect.fail("Server error - API may be
unavailable");",
" }",
" ",
" // For other errors, log but don't fail the test suite",
" console.warn("Unexpected response format or parsing
error");",
" }");",
" }",
" ",
" // =====",
" // SECURITY & HEADERS VALIDATION",
" // =====",
" ",
" // Test 11: Security Headers Check (Optional but recommended)",
" pm.test("Response has security considerations", function () {",
" // This is informational - we check but don't fail",
" const hasSecurityHeaders = pm.response.headers.has("X-
Content-Type-Options") ||",
" pm.response.headers.has("X-Frame-
Options") ||",
" pm.response.headers.has("Strict-
Transport-Security");",
" ",
" if (!hasSecurityHeaders) {",
" console.info("Info: No common security headers
detected");",
" }",
" pm.expect(true).to.be.true; // Always pass",
" });",
" // Test 12: SQL injection (For testing add {{payload}} to
parameters like \\v1\\items\\code={{payload}})",
" pm.test("Check SQL error messages", function () {",

```

```

 const body = pm.response.text().toLowerCase();",
 const sqlErrors = ["sql", "mysql", "syntax error",
 "driver", "database", "query"];",
 sqlErrors.forEach(error => {",
 pm.expect(body).to.not.include(error);",
 });",
 });",
 "",
 "pm.test("Response time normal (Time-based Blind SQL
Injection)", function () {",
 pm.expect(pm.response.responseTime).to.be.below(2000); ",
 });",
 "",
 "// Test: Error messages disclosure",
 "pm.test("Error message is generic and safe", function () {",
 const responseBody = pm.response.text();",
 const sensitiveTerms = ["SQL", "stack trace", "syntax
error at", "fatal error", "uncaught exception", "on line [0-9]",
 "original_query", "exception", "database"];",
 sensitiveTerms.forEach(term => {",
 pm.expect(responseBody.toLowerCase()).to.not.include(term.toLowerCase());",
 });",
 });",
 "",
 "pm.test("Response should not contain sensitive data", function
 () {",
 pm.expect(pm.response.text()).to.not.include("item_name");",
 });",
 "",
 "// Test xx: Information Disclosure (DISABLED)",
 "pm.test("Should not reveal server technology in headers",
function () {",
 const headersToHide = [",
 "X-AspNet-Version",",
 "X-AspNetMvc-Version",",
 "X-Powered-By",",
 "Server",",
];",
 headersToHide.forEach(header => {",
 const value = pm.response.headers.get(header);",
 pm.expect(value, `Header ${header}
found!`).to.be.oneOf([undefined, ""]);",
 });",
 });",
 "",
 "// =====",
 "// SUMMARY LOGGING",
 "// =====",

```

```
 "",
 "console.log(\"=====\");",
 "console.log(\"Test Summary for:\", pm.info.requestName);",
 "console.log(\"Status Code:\", pm.response.code);",
 "console.log(\"Response Time:\", pm.response.responseTime +
\"ms\");",
 "console.log(\"Response Size:\", pm.response.size().body + \"
bytes\");",
 "console.log(\"=====\");"
]
}
},
],
"variable": [
 {
 "key": "baseUrl",
 "value": "https://login.directo.ee/apidirect"
 },
 {
 "key": "version",
 "value": "1"
 },
 {
 "key": "payload",
 "value": ""
 }
]
}
```

Save the file as **collection.json** (File → Save As, choose „All Files“ as the file type so it doesn't save with a .txt extension).

## Importing the file into Postman

1. Open Postman.
2. Click **Import** (top left corner).
3. Choose **File** and select the **collection.json** file you just saved.
4. Click **Import**.

The collection now appears in the left sidebar under **Collections**.

From:  
<https://wiki.directo.ee/> - **Directo Help**

Permanent link:  
[https://wiki.directo.ee/et/api\\_direct/postman?rev=1785306263](https://wiki.directo.ee/et/api_direct/postman?rev=1785306263)

Last update: **2026/07/29 09:24**



