

---

# **StoreKeeper Documentation**

***Release 0.1.0***

**Andras Tim**

November 05, 2015



<b>1</b>	<b>Features</b>	<b>3</b>
<b>2</b>	<b>Demo site</b>	<b>5</b>
<b>3</b>	<b>Languages</b>	<b>7</b>
<b>4</b>	<b>The Guide</b>	<b>9</b>
4.1	Setup . . . . .	9
4.2	Client . . . . .	9
4.3	Server . . . . .	9
<b>5</b>	<b>Indices and tables</b>	<b>65</b>
	<b>HTTP Routing Table</b>	<b>67</b>



StoreKeeper is an open source, multilingual warehouse/store management software

**Warning:** *Project is under development!*



---

# Features

---

The following features are available:

- User login, logout
- Manage items and its barcodes, units, vendors
- Add/remove items in store
- Able to use barcode reader for collecting items
- Can use barcode printer to create labels

The following features are only available via API now:

- Manage users, handle permissions
- Can make acquisition and stocktaking
- Can crate works and its customers (handle outbound, returned items)





---

### Demo site

---

**You can test the latest development version on our demo server:** <http://storekeeper-demo.dras.hu/storekeeper>  
Default username and password: **admin / admin**



---

# Languages

---

StoreKeeper is building in multilingual approach, therefore it can easily translate to a new language.

- You should create a new `.po` file in your language, then put to `client/po/` directory and create a pull request.
- Or use *Transifex* for online translation: <https://www.transifex.com/projects/p/storekeeper/>



## 4.1 Setup

### 4.1.1 Prepare & Run

All commands can run with `package.sh` scripts what you want. All of these scripts has `--help` argument for available getting some info about the current module.

1. Install all dependencies: `./package.sh -p install`
2. Make default configuration files: `./package.sh make_defaults`
3. Check the `config/config.yml` for database and other settings
4. Prepare database: `./package.sh create_database`
5. Start server: `./package.sh start`

Now, you can open the WebUI: <http://localhost:8000/storekeeper>

### 4.1.2 Details

Basically, the `install` command consists of `preinstall` and `postinstall` parts.

- `preinstall` checks/prepares system components for `postinstall` and `start`. This command has only one dependency, an **DEB** based system (for *apt-get install*).
- `postinstall` checks/prepares external dependencies (e.g. Python, Bower modules).

You can modify installing method with this arguments:

- `--global` makes changes on system instead of virtual environments.
- `--production` installs dependencies for production running only (e.g. did not install unit test framework)

## 4.2 Client

The UI of **StoreKeeper** will be implemented in [AngularStrap](#).

## 4.3 Server

The backend of **StoreKeeper** is based on [Flask](#) and used several plugins of it.

### 4.3.1 Interfaces

#### Admin page

**StoreKeeper** has admin interface (with [Flask-Admin](#)), where you can manage database tables and static files directly.

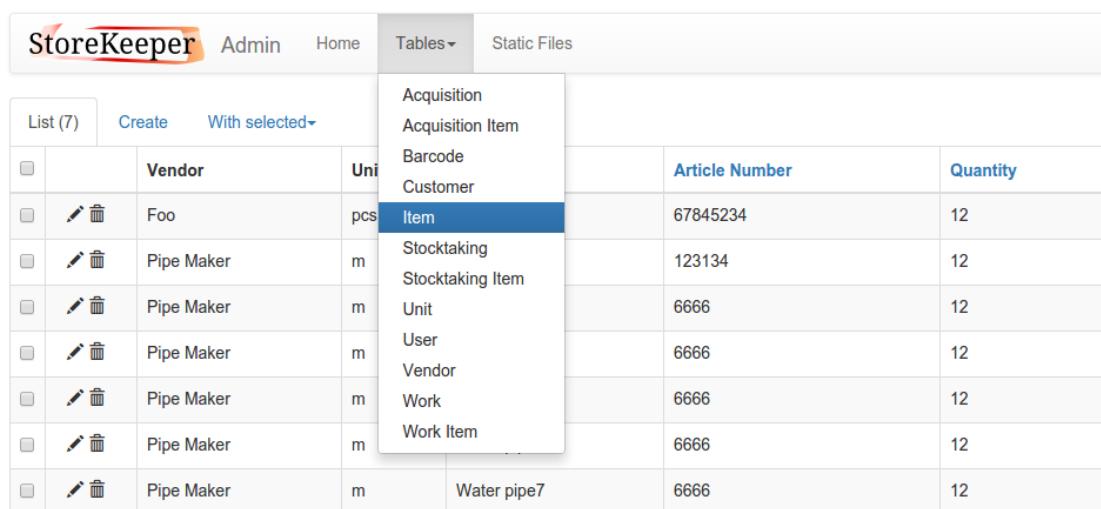
You can reach this on: <http://localhost:8000/<name>/admin>

---

**Note:** Admin page is not available in in production mode.

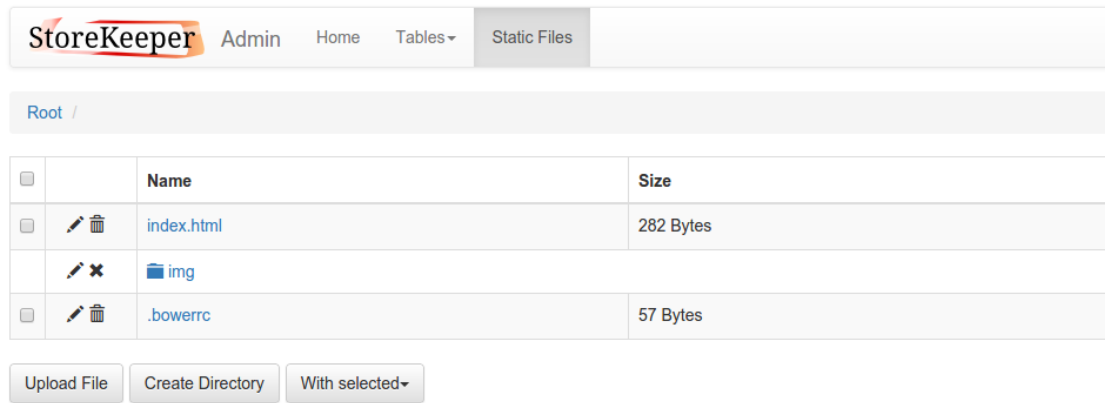
---

#### Interface



The screenshot shows the StoreKeeper Admin interface. At the top, there is a navigation bar with the StoreKeeper logo and links for Admin, Home, Tables (selected), and Static Files. Below the navigation bar, there is a table with columns: Vendor, Unit, Article Number, and Quantity. The table contains 7 rows of data. A dropdown menu is open under the 'Tables' tab, showing a list of tables: Acquisition, Acquisition Item, Barcode, Customer, Item (selected), Stocktaking, Stocktaking Item, Unit, User, Vendor, Work, and Work Item.

Vendor	Unit	Article Number	Quantity
Foo	pcs	67845234	12
Pipe Maker	m	123134	12
Pipe Maker	m	6666	12
Pipe Maker	m	6666	12
Pipe Maker	m	6666	12
Pipe Maker	m	6666	12
Pipe Maker	m	6666	12



## RPC API

**StoreKeeper** uses [HATEOAS](#) RCP API for communicate server and client side each other. The URLs are prefixed with name of application what you can customize in `server/config.yml`.

*Example URL:*

```
http://localhost:8000/<name>/api/<command>
```

## Endpoints

**Acquisitions**   API endpoint for manage acquisitions.

### Data management

#### /api/acquisitions

**GET /storekeeper/api/acquisitions**

List acquisitions

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in

**Example request:**

```
GET /storekeeper/api/acquisitions HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "comment": "Maybe missing some items",
    "id": 1,
    "timestamp": "2015-11-05T10:12:24.804914+00:00"
  },
  {
    "comment": null,
    "id": 2,
    "timestamp": "2015-11-05T10:12:24.804995+00:00"
  }
]
```

**POST /storekeeper/api/acquisitions**

Create acquisition

**Status Codes**

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 422 Unprocessable Entity – there is wrong type / missing field

**Example request:**

```
POST /storekeeper/api/acquisitions HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "comment": "Maybe missing some items"
}
```

**Example response:**

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "comment": "Maybe missing some items",
  "id": 1,
  "timestamp": "2015-11-05T10:12:24.804914+00:00"
}
```

**/api/acquisitions/<id>**

**GET /storekeeper/api/acquisitions/ (int: id)**

Get acquisition

**Query Parameters**

- **id** – ID of selected acquisition for get

**Status Codes**

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no acquisition



**Example request:**

```
GET /storekeeper/api/acquisitions/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "comment": "Maybe missing some items",
  "id": 1,
  "timestamp": "2015-11-05T10:12:24.804914+00:00"
}
```

**PUT /storekeeper/api/acquisitions/ (int: id)**

Update acquisition

**Query Parameters**

- **id** – ID of selected acquisition for put

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no acquisition
- **422 Unprocessable Entity** – there is wrong type / missing field

**Example request:**

```
PUT /storekeeper/api/acquisitions/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "comment": "A box has been damaged"
}
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "comment": "A box has been damaged",
  "id": 1,
  "timestamp": "2015-11-05T10:12:24.804914+00:00"
}
```

**DELETE /storekeeper/api/acquisitions/ (int: id)**

Delete acquisition

**Query Parameters**

- **id** – ID of selected acquisition for delete

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no acquisition

**Example request:**

```
DELETE /storekeeper/api/acquisitions/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**/api/acquisitions/<id>/items**

**GET /storekeeper/api/acquisitions/ (int: id) /items**  
List acquisition items

**Query Parameters**

- **id** – ID of acquisition

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no item

**Example request:**

```
GET /storekeeper/api/acquisitions/1/items HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "id": 1,
    "item": {
      "article_number": "FO213546",
      "id": 2,
      "name": "Pipe",
      "quantity": 32.6,
      "unit": {
        "id": 1,
        "unit": "m"
      },
      "vendor": {
        "id": 2,
        "name": "Star Shop Ltd."
      },
      "warning_quantity": 0.0
    },
    "quantity": 132.4
  },
  {
    "id": 2,
    "item": {
      "article_number": "SK132465",
      "id": 1,
```

```

        "name": "Spray",
        "quantity": 132.3,
        "unit": {
            "id": 2,
            "unit": "pcs"
        },
        "vendor": {
            "id": 1,
            "name": "Heavy Duty Ltd."
        },
        "warning_quantity": 4.0
    },
    "quantity": 32.1
}
]

```

**POST /storekeeper/api/acquisitions/ (int: id) /items**

Create acquisition item

#### Query Parameters

- **id** – ID of acquisition

#### Status Codes

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no item
- 422 Unprocessable Entity – there is wrong type / missing field / can not add one item twice

#### Example request:

```

POST /storekeeper/api/acquisitions/1/items HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
    "item": {
        "article_number": "FO213546",
        "id": 2,
        "name": "Pipe",
        "quantity": 32.6,
        "unit": {
            "id": 1,
            "unit": "m"
        },
        "vendor": {
            "id": 2,
            "name": "Star Shop Ltd."
        },
        "warning_quantity": 0.0
    },
    "quantity": 132.4
}

```

#### Example response:

```

HTTP/1.0 201 CREATED
Content-Type: application/json

{
    "id": 1,

```

```
"item": {
  "article_number": "FO213546",
  "id": 2,
  "name": "Pipe",
  "quantity": 32.6,
  "unit": {
    "id": 1,
    "unit": "m"
  },
  "vendor": {
    "id": 2,
    "name": "Star Shop Ltd."
  },
  "warning_quantity": 0.0
},
"quantity": 132.4
}
```

#### `/api/acquisitions/<id>/items`

**GET** `/storekeeper/api/acquisitions/ (int: id) /items/`  
**int:** *item\_id* Get acquisition item

##### Query Parameters

- **id** – ID of acquisition
- **item\_id** – ID of selected acquisition item for get

##### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no acquisition item

##### Example request:

```
GET /storekeeper/api/acquisitions/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

##### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
}
```

```
"quantity": 132.4
}
```

**PUT** `/storekeeper/api/acquisitions/ (int: id) /items/`  
**int:** *item\_id* Update acquisition item

#### Query Parameters

- **id** – ID of acquisition
- **item\_id** – ID of selected acquisition item for get

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no acquisition item
- **422 Unprocessable Entity** – there is wrong type / missing field / can not add one item twice

#### Example request:

```
PUT /storekeeper/api/acquisitions/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "quantity": 132.4
}
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
```

```
        "name": "Star Shop Ltd.",
      },
      "warning_quantity": 0.0
    },
    "quantity": 132.4
  }
}
```

**DELETE** `/storekeeper/api/acquisitions/ (int: id) /items/`  
`int: item_id` Delete acquisition item

#### Query Parameters

- `id` – ID of acquisition
- `item_id` – ID of selected acquisition item for get

#### Status Codes

- `200 OK` – no error
- `401 Unauthorized` – user was not logged in
- `404 Not Found` – there is no acquisition item

#### Example request:

```
DELETE /storekeeper/api/acquisitions/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**Barcodes** API endpoint for fetching barcodes.

## Data management

### `/api/barcodes`

**GET** `/storekeeper/api/barcodes`  
List barcodes items

#### Status Codes

- `200 OK` – no error
- `401 Unauthorized` – user was not logged in

#### Example request:

```
GET /storekeeper/api/barcodes HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
```

```

    "barcode": "SK-TEST-2370196",
    "id": 1,
    "item_id": 1,
    "main": true,
    "master": true,
    "quantity": 32.7
  },
  {
    "barcode": "9843-184125",
    "id": 2,
    "item_id": 1,
    "main": false,
    "master": false,
    "quantity": 1.5
  }
]

```

**Config** API endpoint for getting client related settings from server config.

### Getting config values

#### /api/config

**GET /storekeeper/api/config**

Get server settings

#### Status Codes

- 200 OK – no error

#### Example request:

```

GET /storekeeper/api/config HTTP/1.1
Host: localhost:8000
Content-Type: application/json

```

#### Example response:

```

HTTP/1.0 200 OK
Content-Type: application/json

{
  "app_name": "storekeeper",
  "app_title": "StoreKeeper",
  "debug": false,
  "forced_language": null
}

```

**Customers** API endpoint for manage customers.

### Data management

#### /api/customers

**GET /storekeeper/api/customers**

List customers

#### Status Codes

- 200 OK – no error

- 401 Unauthorized – user was not logged in

### Example request:

```
GET /storekeeper/api/customers HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "id": 1,
    "name": "Big Customer Ltd."
  },
  {
    "id": 2,
    "name": "Buy Everything Co."
  }
]
```

## POST /storekeeper/api/customers

Create customer

### Status Codes

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 422 Unprocessable Entity – there is wrong type / missing field / customer is already exist

### Example request:

```
POST /storekeeper/api/customers HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "name": "Big Customer Ltd."
}
```

### Example response:

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "id": 1,
  "name": "Big Customer Ltd."
}
```

## /api/customers/<id>

### GET /storekeeper/api/customers/ (int: id)

Get customer

### Query Parameters

- **id** – ID of selected customer for get

### Status Codes



- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no customer

**Example request:**

```
GET /storekeeper/api/customers/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "name": "Big Customer Ltd."
}
```

**PUT /storekeeper/api/customers/ (int: id)**

Update customer

**Query Parameters**

- **id** – ID of selected customer for put

**Status Codes**

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no customer
- 422 Unprocessable Entity – there is wrong type / missing field / customer is already exist

**Example request:**

```
PUT /storekeeper/api/customers/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "name": "new_foo"
}
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "name": "new_foo"
}
```

**DELETE /storekeeper/api/customers/ (int: id)**

Delete customer

**Query Parameters**

- **id** – ID of selected customer for delete

**Status Codes**

- 200 OK – no error

- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no customer

### Example request:

```
DELETE /storekeeper/api/customers/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**Error** API endpoint for getting client side errors to alerting administrator.

## Push error messages

### /api/error

#### POST /storekeeper/api/error

Receive client side errors and forward to logfile / email / syslog (what specified in config)

#### Status Codes

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 422 Unprocessable Entity – there is wrong type / missing field

### Example request:

```
POST /storekeeper/api/error HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "message": "foo is not defined",
  "name": "ReferenceError",
  "stack": "ReferenceError: foo is not defined\n    at Scope.printLabel (http://.../storekeeper/api/error:1:1)"
}
```

### Example response:

```
HTTP/1.0 201 CREATED
Content-Type: application/json

null
```

**Items** API endpoint for manage items.

## Data management

## /api/items

### GET /storekeeper/api/items

List items

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in

#### Example request:

```
GET /storekeeper/api/items HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "article_number": "SK132465",
    "id": 1,
    "name": "Spray",
    "quantity": 132.3,
    "unit": {
      "id": 2,
      "unit": "pcs"
    },
    "vendor": {
      "id": 1,
      "name": "Heavy Duty Ltd."
    },
    "warning_quantity": 4.0
  },
  {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  }
]
```

### POST /storekeeper/api/items

Create item

#### Status Codes

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 422 Unprocessable Entity – there is wrong type / missing field

#### Example request:

```
POST /storekeeper/api/items HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "article_number": "sk132465",
  "name": "Spray",
  "quantity": 132.3,
  "unit": {
    "id": 2,
    "unit": "pcs"
  },
  "vendor": {
    "id": 1,
    "name": "Heavy Duty Ltd."
  },
  "warning_quantity": 4.0
}
```

**Example response:**

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "article_number": "SK132465",
  "id": 1,
  "name": "Spray",
  "quantity": 132.3,
  "unit": {
    "id": 2,
    "unit": "pcs"
  },
  "vendor": {
    "id": 1,
    "name": "Heavy Duty Ltd."
  },
  "warning_quantity": 4.0
}
```

**/api/items/search**

**GET /storekeeper/api/items/search**

Search in items and barcodes

**Status Codes**

- 200 OK – no error
- 401 Unauthorized – user was not logged in

**Example request:**

```
GET /storekeeper/api/items/search?expression=sk&limit=6 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
```

```

    "barcode": "SK586936",
    "item_id": 326,
    "quantity": 1,
    "type": "barcode"
  },
  {
    "article_number": "SE180826",
    "item_id": 309,
    "name": "Skate",
    "type": "item",
    "vendor": "Import"
  }
]

```

**/api/items/<id>**

**GET /storekeeper/api/items/ (int: id)**

Get item

#### Query Parameters

- **id** – ID of selected item for get

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no item

**Example request:**

```

GET /storekeeper/api/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

```

**Example response:**

```

HTTP/1.0 200 OK
Content-Type: application/json

{
  "article_number": "SK132465",
  "id": 1,
  "name": "Spray",
  "quantity": 132.3,
  "unit": {
    "id": 2,
    "unit": "pcs"
  },
  "vendor": {
    "id": 1,
    "name": "Heavy Duty Ltd."
  },
  "warning_quantity": 4.0
}

```

**PUT /storekeeper/api/items/ (int: id)**

Update item

#### Query Parameters

- **id** – ID of selected item for put

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no item
- 422 Unprocessable Entity – there is wrong type / missing field

### Example request:

```
PUT /storekeeper/api/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "article_number": "sk132465",
  "name": "Spray",
  "quantity": 132.3,
  "unit": {
    "id": 2,
    "unit": "pcs"
  },
  "vendor": {
    "id": 1,
    "name": "Heavy Duty Ltd."
  },
  "warning_quantity": 4.0
}
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "article_number": "SK132465",
  "id": 1,
  "name": "Spray",
  "quantity": 132.3,
  "unit": {
    "id": 2,
    "unit": "pcs"
  },
  "vendor": {
    "id": 1,
    "name": "Heavy Duty Ltd."
  },
  "warning_quantity": 4.0
}
```

**DELETE** /storekeeper/api/items/ (int: *id*)

Delete item

### Query Parameters

- **id** – ID of selected item for delete

### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no item

### Example request:

```
DELETE /storekeeper/api/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**/api/items/<id>/barcodes**

**GET /storekeeper/api/items/ (int: id) /barcodes**  
List barcodes.

**Query Parameters**

- **id** – ID of item

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no item

**Example request:**

```
GET /storekeeper/api/items/1/barcodes HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "barcode": "SK-TEST-6834975",
    "id": 1,
    "main": true,
    "master": true,
    "quantity": 32.7
  },
  {
    "barcode": "9843-184125",
    "id": 2,
    "main": false,
    "master": false,
    "quantity": 1.5
  }
]
```

**POST /storekeeper/api/items/ (int: id) /barcodes**  
Create barcode (if missing barcode then server will generate one)

**Query Parameters**

- **id** – ID of item

**Status Codes**

- **201 Created** – no error

- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no item
- 422 Unprocessable Entity – there is wrong type / missing field / can not add one barcode twice / can not generate unique new barcode / can not set non-main barcode as master barcode / can not set more than one master barcode to an item

### Example request:

```
POST /storekeeper/api/items/1/barcodes HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "barcode": "SK-TEST-6834975",
  "master": true,
  "quantity": 32.7
}
```

### Example response:

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "barcode": "SK-TEST-6834975",
  "id": 1,
  "main": true,
  "master": true,
  "quantity": 32.7
}
```

## /api/items/<id>/barcodes/<id>

**GET /storekeeper/api/items/ (int: item\_id) /barcodes/**  
**int: id** Get barcode

### Query Parameters

- **id** – ID of selected barcode for get
- **item\_id** – ID of item

### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no barcode

### Example request:

```
GET /storekeeper/api/items/1/barcodes/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "barcode": "SK-TEST-6834975",
  "id": 1,
  "main": true,
```



```

    "master": true,
    "quantity": 32.7
  }

```

**PUT** `/storekeeper/api/items/ (int: item_id) /barcodes/`  
**int: id** Update barcode

#### Query Parameters

- **id** – ID of selected barcode for put
- **item\_id** – ID of item

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no barcode
- **422 Unprocessable Entity** – there is wrong type / missing field / can not add one barcode twice / can not set non-main barcode as master barcode / can not set more than one master barcode to an item

#### Example request:

```

PUT /storekeeper/api/items/1/barcodes/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "barcode": "SK-TEST-6834975",
  "master": true,
  "quantity": 32.7
}

```

#### Example response:

```

HTTP/1.0 200 OK
Content-Type: application/json

{
  "barcode": "SK-TEST-6834975",
  "id": 1,
  "main": true,
  "master": true,
  "quantity": 32.7
}

```

**DELETE** `/storekeeper/api/items/ (int: item_id) /barcodes/`  
**int: id** Delete barcode

#### Query Parameters

- **id** – ID of selected barcode for delete
- **item\_id** – ID of item

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no barcode

#### Example request:

```
DELETE /storekeeper/api/items/1/barcodes/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

## Commands

### /api/items/<id>/barcodes/<id>/print

**GET** /storekeeper/api/items/ (int: *item\_id*) /barcodes/  
int: *id*/**print** Generate barcode label to PDF with some details, and starts downloading that.

#### Query Parameters

- **id** – ID of selected barcode for get
- **item\_id** – ID of item

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no barcode

### Example request:

```
GET /storekeeper/api/items/1/barcodes/1/print HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/pdf
Content-Disposition: attachment; filename=label__SK642031__4f0ff51c73703295643a325e55bc7e
Content-Length: 11234

<file content>
```

**PUT** /storekeeper/api/items/ (int: *item\_id*) /barcodes/  
int: *id*/**print** Print barcode label with some details

#### Query Parameters

- **id** – ID of selected barcode for get
- **item\_id** – ID of item

#### Status Codes

- 200 OK – no error
- 400 Bad Request – missing pycups python3 module
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no barcode
- 422 Unprocessable Entity – there is wrong type / missing field

**Example request:**

```
PUT /storekeeper/api/items/1/barcodes/1/print HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "copies": 3
}
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**Session** API endpoint for manage the current session.

**Session management****/api/session****GET /storekeeper/api/session**

Get current session

**Status Codes**

- 200 OK – no error
- 401 Unauthorized – user was not logged in

**Example request:**

```
GET /storekeeper/api/session HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "admin": true,
  "disabled": false,
  "email": "admin@test.com",
  "id": 1,
  "username": "admin"
}
```

**POST /storekeeper/api/session**

Login user

**Status Codes**

- 201 Created – no error
- 401 Unauthorized – bad authentication data or user is disabled
- 422 Unprocessable Entity – there is wrong type / missing field

**Example request:**

```
POST /storekeeper/api/session HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "password": "secret",
  "remember": false,
  "username": "admin"
}
```

### Example response:

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "admin": true,
  "disabled": false,
  "email": "admin@test.com",
  "id": 1,
  "username": "admin"
}
```

**DELETE /storekeeper/api/session**  
Logout user

### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in

### Example request:

```
DELETE /storekeeper/api/session HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**Stocktakings** API endpoint for manage stocktaking results.

## Data management

### /api/stocktakings

**GET /storekeeper/api/stocktakings**  
List stocktakings

### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in

### Example request:

```
GET /storekeeper/api/stocktakings HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "close_timestamp": null,
    "close_user": {
      "admin": null,
      "disabled": null,
      "email": null,
      "id": null,
      "username": null
    },
    "comment": "Maybe missing some items",
    "id": 1,
    "timestamp": "2015-11-05T10:12:24.805056+00:00"
  },
  {
    "close_timestamp": null,
    "close_user": {
      "admin": null,
      "disabled": null,
      "email": null,
      "id": null,
      "username": null
    },
    "comment": null,
    "id": 2,
    "timestamp": "2015-11-05T10:12:24.805090+00:00"
  }
]
```

**POST /storekeeper/api/stocktakings**

Create stocktaking

**Status Codes**

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 422 Unprocessable Entity – there is wrong type / missing field

**Example request:**

```
POST /storekeeper/api/stocktakings HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "comment": "Maybe missing some items"
}
```

**Example response:**

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
```

```
"close_timestamp": null,
"close_user": {
  "admin": null,
  "disabled": null,
  "email": null,
  "id": null,
  "username": null
},
"comment": "Maybe missing some items",
"id": 1,
"timestamp": "2015-11-05T10:12:24.805056+00:00"
}
```

### `/api/stocktakings/<id>`

**GET** `/storekeeper/api/stocktakings/` (int: *id*)

Get stocktaking

#### Query Parameters

- **id** – ID of selected stocktaking for get

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no stocktaking

**Example request:**

```
GET /storekeeper/api/stocktakings/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "close_timestamp": null,
  "close_user": {
    "admin": null,
    "disabled": null,
    "email": null,
    "id": null,
    "username": null
  },
  "comment": "Maybe missing some items",
  "id": 1,
  "timestamp": "2015-11-05T10:12:24.805056+00:00"
}
```

**PUT** `/storekeeper/api/stocktakings/` (int: *id*)

Update stocktaking

#### Query Parameters

- **id** – ID of selected stocktaking for put

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in

- 404 Not Found – there is no stocktaking
- 422 Unprocessable Entity – there is wrong type / missing field

**Example request:**

```
PUT /storekeeper/api/stocktakings/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "comment": "A box has been damaged"
}
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "close_timestamp": null,
  "close_user": {
    "admin": null,
    "disabled": null,
    "email": null,
    "id": null,
    "username": null
  },
  "comment": "A box has been damaged",
  "id": 1,
  "timestamp": "2015-11-05T10:12:24.805056+00:00"
}
```

**DELETE /storekeeper/api/stocktakings/ (int: id)**

Delete stocktaking

**Query Parameters**

- **id** – ID of selected stocktaking for delete

**Status Codes**

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no stocktaking

**Example request:**

```
DELETE /storekeeper/api/stocktakings/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**/api/stocktakings/<id>/items**

**GET /storekeeper/api/stocktakings/ (int: id) /items**

List stocktaking items.

**Query Parameters**

- **id** – ID of stocktaking

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no item

#### Example request:

```
GET /storekeeper/api/stocktakings/1/items HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "id": 1,
    "item": {
      "article_number": "FO213546",
      "id": 2,
      "name": "Pipe",
      "quantity": 32.6,
      "unit": {
        "id": 1,
        "unit": "m"
      },
      "vendor": {
        "id": 2,
        "name": "Star Shop Ltd."
      },
      "warning_quantity": 0.0
    },
    "quantity": 132.2
  },
  {
    "id": 2,
    "item": {
      "article_number": "SK132465",
      "id": 1,
      "name": "Spray",
      "quantity": 132.3,
      "unit": {
        "id": 2,
        "unit": "pcs"
      },
      "vendor": {
        "id": 1,
        "name": "Heavy Duty Ltd."
      },
      "warning_quantity": 4.0
    },
    "quantity": 32.3
  }
]
```

**POST /storekeeper/api/stocktakings/(int: id)/items**  
Create stocktaking item

#### Query Parameters



- **id** – ID of stocktaking

#### Status Codes

- **201 Created** – no error
- **401 Unauthorized** – user was not logged in
- **403 Forbidden** – can not add new stocktakings item after items was closed
- **404 Not Found** – there is no item
- **422 Unprocessable Entity** – there is wrong type / missing field / can not add one item twice

#### Example request:

```
POST /storekeeper/api/stocktakings/1/items HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "quantity": 132.2
}
```

#### Example response:

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "id": 1,
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "quantity": 132.2
}
```

/api/stocktakings/<id>/items/<item\_id>

**GET** /storekeeper/api/stocktakings/ (int: id) /items/  
int: item\_id Get stocktaking item

#### Query Parameters

- **id** – ID of stocktaking
- **item\_id** – ID of selected stocktaking item for get

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no stocktaking item

#### Example request:

```
GET /storekeeper/api/stocktakings/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "quantity": 132.2
}
```

**PUT** /storekeeper/api/stocktakings/ (int: id) /items/  
int: item\_id Update stocktaking item

#### Query Parameters

- **id** – ID of stocktaking
- **item\_id** – ID of selected stocktaking item for put

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **403 Forbidden** – can not change work item after outbound/returned items was closed
- **404 Not Found** – there is no stocktaking item

- 422 Unprocessable Entity – there is wrong type / missing field / can not add one item twice

**Example request:**

```
PUT /storekeeper/api/stocktakings/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "quantity": 132.2
}
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "quantity": 132.2
}
```

**DELETE** /storekeeper/api/stocktakings/ (int: id) /items/  
 int: item\_id Delete stocktaking item

**Query Parameters**

- **id** – ID of stocktaking
- **item\_id** – ID of selected stocktaking item for delete

**Status Codes**

- 200 OK – no error

- 401 Unauthorized – user was not logged in
- 403 Forbidden – can not delete stocktaking item after items was closed
- 404 Not Found – there is no stocktaking item

### Example request:

```
DELETE /storekeeper/api/stocktakings/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

## Commands

### /api/stocktakings/<id>/close

**PUT /storekeeper/api/stocktakings/ (int: id) /close**  
Close items on stocktaking

#### Query Parameters

- **id** – ID of selected stocktaking for put

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no stocktaking

### Example request:

```
PUT /storekeeper/api/stocktakings/1/close HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "close_timestamp": "2015-11-05T10:12:24.805114+00:00",
  "close_user": {
    "admin": false,
    "disabled": false,
    "email": "foo@bar.com",
    "id": 2,
    "username": "foo"
  },
  "comment": "Maybe missing some items",
  "id": 1,
  "timestamp": "2015-11-05T10:12:24.805056+00:00"
}
```

**Units** API endpoint for manage units.

## Data management

### /api/units

**GET /storekeeper/api/units**

List units

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in

#### Example request:

```
GET /storekeeper/api/units HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "id": 1,
    "unit": "m"
  },
  {
    "id": 2,
    "unit": "pcs"
  }
]
```

**POST /storekeeper/api/units**

Create unit

#### Status Codes

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 422 Unprocessable Entity – there is wrong type / missing field / unit is already exist

#### Example request:

```
POST /storekeeper/api/units HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "unit": "m"
}
```

#### Example response:

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "id": 1,
  "unit": "m"
}
```

**/api/units/<id>**

**GET /storekeeper/api/units/ (int: id)**  
Get unit

**Query Parameters**

- **id** – ID of selected unit for get

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no unit

**Example request:**

```
GET /storekeeper/api/units/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "unit": "m"
}
```

**PUT /storekeeper/api/units/ (int: id)**  
Update unit

**Query Parameters**

- **id** – ID of selected unit for put

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no unit
- **422 Unprocessable Entity** – there is wrong type / missing field / unit is already exist

**Example request:**

```
PUT /storekeeper/api/units/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "unit": "dl"
}
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "unit": "dl"
}
```

**DELETE /storekeeper/api/units/ (int: id)**

Delete unit

**Query Parameters**

- **id** – ID of selected unit for delete

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no unit

**Example request:**

```
DELETE /storekeeper/api/units/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**Users** API endpoint for manage users.**Data management****/api/users****GET /storekeeper/api/users**List users (*for administrators only*)**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **403 Forbidden** – user has not enough rights

**Example request:**

```
GET /storekeeper/api/users HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "admin": true,
    "disabled": false,
    "email": "admin@test.com",
    "id": 1,
    "username": "admin"
  },
  {
    "admin": false,
```

```
"disabled": false,
"email": "foo@bar.com",
"id": 2,
"username": "foo"
}
]
```

#### POST /storekeeper/api/users

Create user (*for administrators only*)

##### Status Codes

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 403 Forbidden – user has not enough rights
- 422 Unprocessable Entity – {original} / user is already exist

##### Example request:

```
POST /storekeeper/api/users HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "email": "foo@bar.com",
  "password": "bar",
  "username": "foo"
}
```

##### Example response:

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "admin": false,
  "disabled": false,
  "email": "foo@bar.com",
  "id": 2,
  "username": "foo"
}
```

#### /api/users/<id>

#### GET /storekeeper/api/users/ (int: id)

Get user

##### Query Parameters

- **id** – ID of selected user for get

##### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no user

##### Example request:

```
GET /storekeeper/api/users/2 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```



**Example response:**

```

HTTP/1.0 200 OK
Content-Type: application/json

{
  "admin": false,
  "disabled": false,
  "email": "foo@bar.com",
  "id": 2,
  "username": "foo"
}

```

**PUT /storekeeper/api/users/ (int: id)**

Update user

**Query Parameters**

- **id** – ID of selected user for put

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **403 Forbidden** – user can not modify other users
- **404 Not Found** – there is no user
- **422 Unprocessable Entity** – {original} / user is already exist

**Example request:**

```

PUT /storekeeper/api/users/2 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "email": "foo@bar.com",
  "password": "bar",
  "username": "new_foo"
}

```

**Example response:**

```

HTTP/1.0 200 OK
Content-Type: application/json

{
  "admin": false,
  "disabled": false,
  "email": "foo@bar.com",
  "id": 2,
  "username": "new_foo"
}

```

**DELETE /storekeeper/api/users/ (int: id)**Delete user (*for administrators only*)**Query Parameters**

- **id** – ID of selected user for delete

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in

- 403 Forbidden – user can not remove itself
- 404 Not Found – there is no user

### Example request:

```
DELETE /storekeeper/api/users/2 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

## Config management

### /api/users/<id>/config

**GET /storekeeper/api/users/ (int: id) /config**  
List user items.

#### Query Parameters

- **id** – ID of user

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no item

### Example request:

```
GET /storekeeper/api/users/2/config HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "name": "lang",
    "value": "hu"
  },
  {
    "name": "fruits",
    "value": "[\"apple\", \"orange\", \"banana\"]"
  }
]
```

**POST /storekeeper/api/users/ (int: id) /config**  
Create user item

#### Query Parameters

- **id** – ID of user

#### Status Codes

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no item
- 422 Unprocessable Entity – there is wrong type / missing field / can not add one item twice

**Example request:**

```
POST /storekeeper/api/users/2/config HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "name": "lang",
  "value": "hu"
}
```

**Example response:**

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "name": "lang",
  "value": "hu"
}
```

**/api/users/<id>/config/<id>**

**GET** /storekeeper/api/users/ (int: id) /config/  
string: name Get user item

**Query Parameters**

- **id** – ID of user
- **name** – Name of selected user config value for get

**Status Codes**

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no user item

**Example request:**

```
GET /storekeeper/api/users/2/config/lang HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "name": "lang",
  "value": "hu"
}
```

**PUT** /storekeeper/api/users/ (int: id) /config/  
string: name Update user item

### Query Parameters

- **id** – ID of user
- **name** – Name of selected user config value for put

### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no user item
- **422 Unprocessable Entity** – there is wrong type / missing field / can not use one config name twice

### Example request:

```
PUT /storekeeper/api/users/2/config/lang HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "name": "lang",
  "value": "hu"
}
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "name": "lang",
  "value": "hu"
}
```

**DELETE /storekeeper/api/users/ (int: id) /config/**  
**string: name** Delete user item

### Query Parameters

- **id** – ID of user
- **name** – Name of selected user config value for delete

### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no user item

### Example request:

```
DELETE /storekeeper/api/users/2/config/lang HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**Vendors** API endpoint for manage vendors.

## Data management

### /api/vendors

#### GET /storekeeper/api/vendors

List vendors

##### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in

##### Example request:

```
GET /storekeeper/api/vendors HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

##### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "id": 1,
    "name": "Heavy Duty Ltd."
  },
  {
    "id": 2,
    "name": "Star Shop Ltd."
  }
]
```

#### POST /storekeeper/api/vendors

Create vendor

##### Status Codes

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 422 Unprocessable Entity – {original} / vendor is already exist

##### Example request:

```
POST /storekeeper/api/vendors HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "name": "Heavy Duty Ltd."
}
```

##### Example response:

```
HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "id": 1,
  "name": "Heavy Duty Ltd."
}
```

## /api/vendors/<id>

**GET** /storekeeper/api/vendors/ (int: id)  
Get vendor

### Query Parameters

- **id** – ID of selected vendor for get

### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no vendor

### Example request:

```
GET /storekeeper/api/vendors/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "name": "Heavy Duty Ltd."
}
```

**PUT** /storekeeper/api/vendors/ (int: id)  
Update vendor

### Query Parameters

- **id** – ID of selected vendor for put

### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no vendor
- **422 Unprocessable Entity** – {original} / vendor is already exist

### Example request:

```
PUT /storekeeper/api/vendors/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "name": "new_foo"
}
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "name": "new_foo"
}
```

**DELETE /storekeeper/api/vendors/ (int: id)**

Delete vendor

#### Query Parameters

- **id** – ID of selected vendor for delete

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no vendor

#### Example request:

```
DELETE /storekeeper/api/vendors/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

**Works** API endpoint for manage works.

### Data management

#### /api/works

**GET /storekeeper/api/works**

List works

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in

#### Example request:

```
GET /storekeeper/api/works HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

[
  {
    "comment": "First work",
    "customer": {
      "id": 1,
      "name": "Big Customer Ltd."
    },
    "id": 1,
    "outbound_close_timestamp": null,
    "outbound_close_user": {
      "admin": null,
```

```
        "disabled": null,
        "email": null,
        "id": null,
        "username": null
    },
    "returned_close_timestamp": null,
    "returned_close_user": {
        "admin": null,
        "disabled": null,
        "email": null,
        "id": null,
        "username": null
    }
},
{
    "comment": null,
    "customer": {
        "id": 2,
        "name": "Buy Everything Co."
    },
    "id": 2,
    "outbound_close_timestamp": null,
    "outbound_close_user": {
        "admin": null,
        "disabled": null,
        "email": null,
        "id": null,
        "username": null
    },
    "returned_close_timestamp": null,
    "returned_close_user": {
        "admin": null,
        "disabled": null,
        "email": null,
        "id": null,
        "username": null
    }
}
]
```

**POST /storekeeper/api/works**

Create work

**Status Codes**

- 201 Created – no error
- 401 Unauthorized – user was not logged in
- 422 Unprocessable Entity – there is wrong type / missing field

**Example request:**

```
POST /storekeeper/api/works HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
    "comment": "First work",
    "customer": {
        "id": 1,
        "name": "Big Customer Ltd."
    }
}
```



**Example response:**

```

HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "comment": "First work",
  "customer": {
    "id": 1,
    "name": "Big Customer Ltd."
  },
  "id": 1,
  "outbound_close_timestamp": null,
  "outbound_close_user": {
    "admin": null,
    "disabled": null,
    "email": null,
    "id": null,
    "username": null
  },
  "returned_close_timestamp": null,
  "returned_close_user": {
    "admin": null,
    "disabled": null,
    "email": null,
    "id": null,
    "username": null
  }
}

```

/api/works/<id>

GET /storekeeper/api/works/ (int: id)

Get work

**Query Parameters**

- **id** – ID of selected work for get

**Status Codes**

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no work

**Example request:**

```

GET /storekeeper/api/works/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

```

**Example response:**

```

HTTP/1.0 200 OK
Content-Type: application/json

{
  "comment": "First work",
  "customer": {
    "id": 1,
    "name": "Big Customer Ltd."
  },
  "id": 1,
  "outbound_close_timestamp": null,

```

```
"outbound_close_user": {
  "admin": null,
  "disabled": null,
  "email": null,
  "id": null,
  "username": null
},
"returned_close_timestamp": null,
"returned_close_user": {
  "admin": null,
  "disabled": null,
  "email": null,
  "id": null,
  "username": null
}
}
```

**PUT** /storekeeper/api/works/ (int: id)

Update work

#### Query Parameters

- **id** – ID of selected work for put

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no work
- **422 Unprocessable Entity** – there is wrong type / missing field

#### Example request:

```
PUT /storekeeper/api/works/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "comment": "Something are not finished",
  "customer": {
    "id": 1,
    "name": "Big Customer Ltd."
  }
}
```

#### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "comment": "Something are not finished",
  "customer": {
    "id": 1,
    "name": "Big Customer Ltd."
  },
  "id": 1,
  "outbound_close_timestamp": null,
  "outbound_close_user": {
    "admin": null,
    "disabled": null,
    "email": null,
    "id": null,
```

```

    "username": null
  },
  "returned_close_timestamp": null,
  "returned_close_user": {
    "admin": null,
    "disabled": null,
    "email": null,
    "id": null,
    "username": null
  }
}

```

**DELETE** /storekeeper/api/works/ (int: *id*)

Delete work

#### Query Parameters

- **id** – ID of selected work for delete

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no work

#### Example request:

```

DELETE /storekeeper/api/works/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

```

#### Example response:

```

HTTP/1.0 200 OK
Content-Type: application/json

null

```

/api/works/<id>/items

**GET** /storekeeper/api/works/ (int: *id*) /items

List work items

#### Query Parameters

- **id** – ID of work

#### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 404 Not Found – there is no item

#### Example request:

```

GET /storekeeper/api/works/1/items HTTP/1.1
Host: localhost:8000
Content-Type: application/json

```

#### Example response:

```

HTTP/1.0 200 OK
Content-Type: application/json

```

```
[
  {
    "id": 1,
    "item": {
      "article_number": "FO213546",
      "id": 2,
      "name": "Pipe",
      "quantity": 32.6,
      "unit": {
        "id": 1,
        "unit": "m"
      },
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "outbound_quantity": 132.8,
  "returned_quantity": null
},
{
  "id": 2,
  "item": {
    "article_number": "SK132465",
    "id": 1,
    "name": "Spray",
    "quantity": 132.3,
    "unit": {
      "id": 2,
      "unit": "pcs"
    },
  },
  "vendor": {
    "id": 1,
    "name": "Heavy Duty Ltd."
  },
  "warning_quantity": 4.0
},
"outbound_quantity": 32.2,
"returned_quantity": 0.0
}
]
```

**POST /storekeeper/api/works/(int: id)/items**

Create work item

#### Query Parameters

- **id** – ID of work

#### Status Codes

- **201 Created** – no error
- **401 Unauthorized** – user was not logged in
- **403 Forbidden** – can not add new work item after outbound items was closed
- **404 Not Found** – there is no item
- **422 Unprocessable Entity** – there is wrong type / missing field / can not add one item twice

**Example request:**

```

POST /storekeeper/api/works/1/items HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "outbound_quantity": 132.8
}

```

**Example response:**

```

HTTP/1.0 201 CREATED
Content-Type: application/json

{
  "id": 1,
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "outbound_quantity": 132.8,
  "returned_quantity": null
}

```

**/api/works/<id>/items/<item\_id>**

**GET /storekeeper/api/works/ (int: id) /items/**  
**int: item\_id** Get work item

**Query Parameters**

- **id** – ID of work
- **item\_id** – ID of selected work item for get

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in

- 404 Not Found – there is no work item

### Example request:

```
GET /storekeeper/api/works/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "outbound_quantity": 132.8,
  "returned_quantity": null
}
```

**PUT /storekeeper/api/works/(int: id) /items/**  
**int: item\_id** Update work item

### Query Parameters

- **id** – ID of work
- **item\_id** – ID of selected work item for get

### Status Codes

- 200 OK – no error
- 401 Unauthorized – user was not logged in
- 403 Forbidden – can not change work item after outbound/returned items was closed
- 404 Not Found – there is no work item
- 422 Unprocessable Entity – there is wrong type / missing field / can not add one item twice

### Example request:

```
PUT /storekeeper/api/works/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

{
  "item": {
    "article_number": "FO213546",
    "id": 2,
    "name": "Pipe",
```

```

    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
    "vendor": {
      "id": 2,
      "name": "Star Shop Ltd."
    },
    "warning_quantity": 0.0
  },
  "outbound_quantity": 132.8
}

```

**Example response:**

```

HTTP/1.0 200 OK
Content-Type: application/json

{
  "id": 1,
  "item": {
    "article_number": "F0213546",
    "id": 2,
    "name": "Pipe",
    "quantity": 32.6,
    "unit": {
      "id": 1,
      "unit": "m"
    },
  },
  "vendor": {
    "id": 2,
    "name": "Star Shop Ltd."
  },
  "warning_quantity": 0.0
},
"outbound_quantity": 132.8,
"returned_quantity": null
}

```

**DELETE /storekeeper/api/works/ (int: id) /items/**  
**int: item\_id** Delete work item

**Query Parameters**

- **id** – ID of work
- **item\_id** – ID of selected work item for get

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **403 Forbidden** – can not delete work item after outbound/returned items was closed
- **404 Not Found** – there is no work item

**Example request:**

```

DELETE /storekeeper/api/works/1/items/1 HTTP/1.1
Host: localhost:8000
Content-Type: application/json

```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

null
```

## Commands

### /api/works/<id>/close-outbound

**PUT /storekeeper/api/works/ (int: id) /close-outbound**

Close outbound items on work

#### Query Parameters

- **id** – ID of selected work for put

#### Status Codes

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no work

### Example request:

```
PUT /storekeeper/api/works/1/close-outbound HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

### Example response:

```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "comment": "First work",
  "customer": {
    "id": 1,
    "name": "Big Customer Ltd."
  },
  "id": 1,
  "outbound_close_timestamp": "2015-11-05T10:12:24.805817+00:00",
  "outbound_close_user": {
    "admin": false,
    "disabled": false,
    "email": "foo@bar.com",
    "id": 2,
    "username": "foo"
  },
  "returned_close_timestamp": null,
  "returned_close_user": {
    "admin": null,
    "disabled": null,
    "email": null,
    "id": null,
    "username": null
  }
}
```



**/api/works/<id>/close-returned****PUT /storekeeper/api/works/ (int: id) /close-returned**

Close returned items on work

**Query Parameters**

- **id** – ID of selected work for put

**Status Codes**

- **200 OK** – no error
- **401 Unauthorized** – user was not logged in
- **404 Not Found** – there is no work

**Example request:**

```
PUT /storekeeper/api/works/1/close-returned HTTP/1.1
Host: localhost:8000
Content-Type: application/json
```

**Example response:**

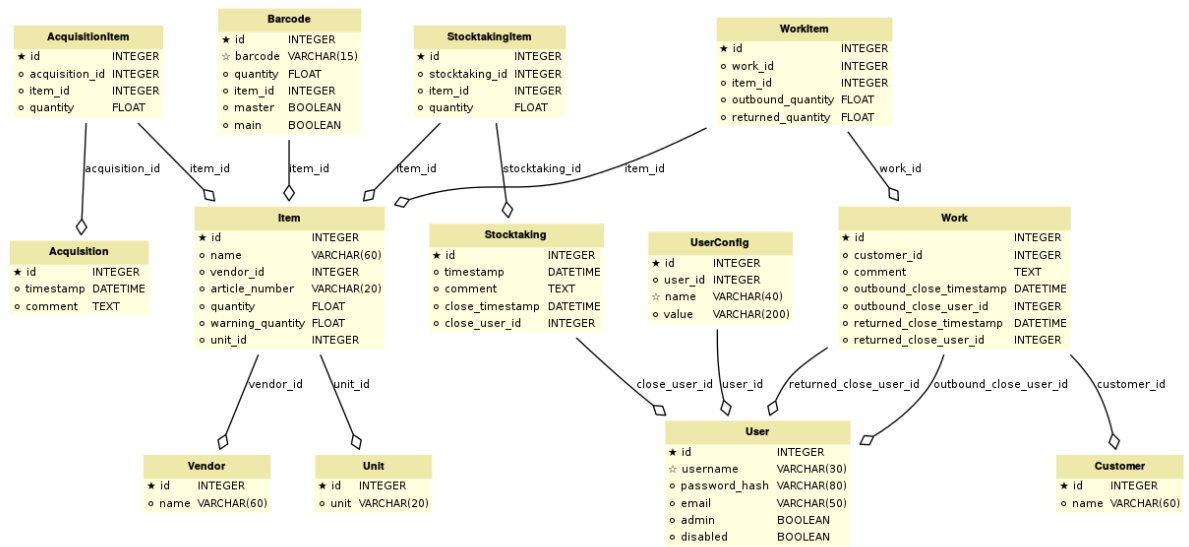
```
HTTP/1.0 200 OK
Content-Type: application/json

{
  "comment": "First work",
  "customer": {
    "id": 1,
    "name": "Big Customer Ltd."
  },
  "id": 1,
  "outbound_close_timestamp": "2015-11-05T10:12:24.805817+00:00",
  "outbound_close_user": {
    "admin": false,
    "disabled": false,
    "email": "foo@bar.com",
    "id": 2,
    "username": "foo"
  },
  "returned_close_timestamp": "2015-11-05T10:12:24.805862+00:00",
  "returned_close_user": {
    "admin": false,
    "disabled": false,
    "email": "foo@bar.com",
    "id": 2,
    "username": "foo"
  }
}
```

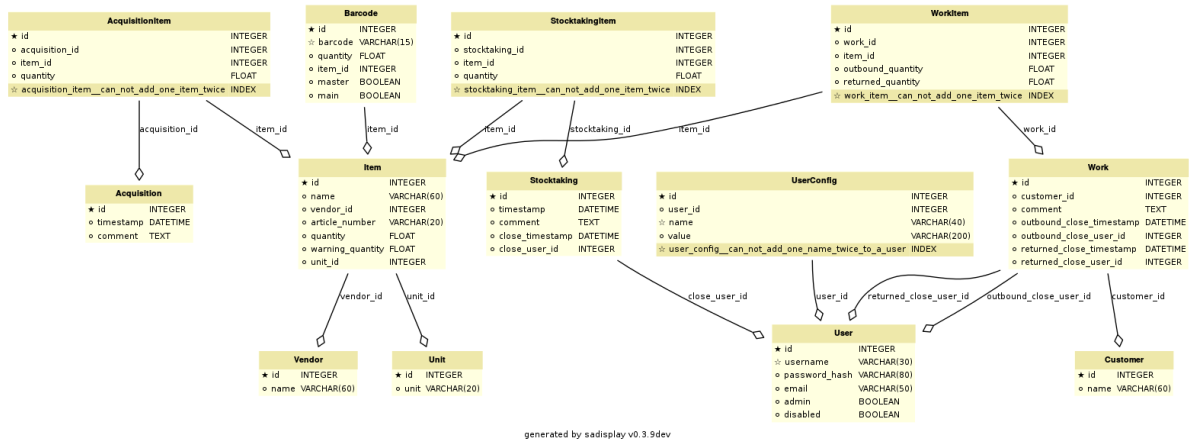
## 4.3.2 Details

### SQL Model

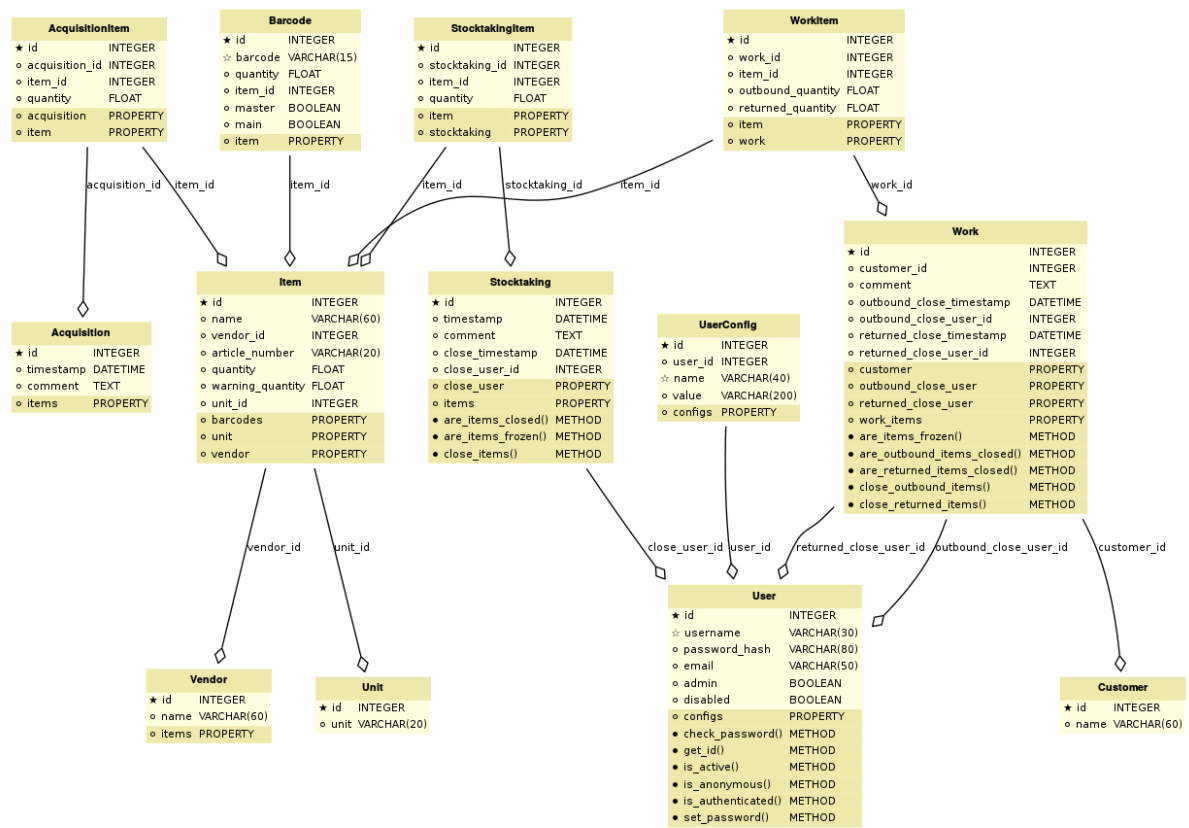
This software is SQL independent (thank you for [SQLAlchemy](#)), and we want to hold it in future. However, requirements, installer scripts, tests were prepared for and ran on PostgreSQL, MySQL and SQLite databases.



## Model with indexes



## Model with active properties and methods





---

## Indices and tables

---

- `genindex`



## /storekeeper

	GET /storekeeper/api/works, <a href="#">51</a>
GET /storekeeper/api/acquisitions, <a href="#">11</a>	GET /storekeeper/api/works/(int:id), <a href="#">53</a>
GET /storekeeper/api/acquisitions/(int:id), <a href="#">12</a>	GET /storekeeper/api/works/(int:id)/items, <a href="#">55</a>
GET /storekeeper/api/acquisitions/(int:id)/items, <a href="#">14</a>	GET /storekeeper/api/works/(int:id)/items/(int:item_id), <a href="#">57</a>
GET /storekeeper/api/acquisitions/(int:id)/items/(int:item_id), <a href="#">16</a>	POST /storekeeper/api/acquisitions, <a href="#">12</a>
GET /storekeeper/api/barcodes, <a href="#">18</a>	POST /storekeeper/api/acquisitions/(int:id)/items, <a href="#">15</a>
GET /storekeeper/api/config, <a href="#">19</a>	POST /storekeeper/api/customers, <a href="#">20</a>
GET /storekeeper/api/customers, <a href="#">19</a>	POST /storekeeper/api/error, <a href="#">22</a>
GET /storekeeper/api/customers/(int:id), <a href="#">20</a>	POST /storekeeper/api/items, <a href="#">23</a>
GET /storekeeper/api/items, <a href="#">23</a>	POST /storekeeper/api/items/(int:id)/barcodes, <a href="#">27</a>
GET /storekeeper/api/items/(int:id), <a href="#">25</a>	POST /storekeeper/api/session, <a href="#">31</a>
GET /storekeeper/api/items/(int:id)/barcodes, <a href="#">27</a>	POST /storekeeper/api/stocktakings, <a href="#">33</a>
GET /storekeeper/api/items/(int:item_id)/barcodes, <a href="#">28</a>	POST /storekeeper/api/stocktakings/(int:id)/items, <a href="#">36</a>
GET /storekeeper/api/items/(int:item_id)/barcodes/(int:id), <a href="#">30</a>	POST /storekeeper/api/units, <a href="#">41</a>
GET /storekeeper/api/items/(int:item_id)/barcodes/(int:id)/print, <a href="#">30</a>	POST /storekeeper/api/users, <a href="#">44</a>
GET /storekeeper/api/items/search, <a href="#">24</a>	POST /storekeeper/api/users/(int:id)/config, <a href="#">46</a>
GET /storekeeper/api/session, <a href="#">31</a>	POST /storekeeper/api/vendors, <a href="#">49</a>
GET /storekeeper/api/stocktakings, <a href="#">32</a>	POST /storekeeper/api/works, <a href="#">52</a>
GET /storekeeper/api/stocktakings/(int:id), <a href="#">34</a>	POST /storekeeper/api/works/(int:id)/items, <a href="#">56</a>
GET /storekeeper/api/stocktakings/(int:id)/items, <a href="#">35</a>	PUT /storekeeper/api/acquisitions/(int:id), <a href="#">13</a>
GET /storekeeper/api/stocktakings/(int:id)/items/(int:item_id), <a href="#">38</a>	PUT /storekeeper/api/acquisitions/(int:id)/items/(int:item_id), <a href="#">17</a>
GET /storekeeper/api/units, <a href="#">41</a>	PUT /storekeeper/api/customers/(int:id), <a href="#">21</a>
GET /storekeeper/api/units/(int:id), <a href="#">42</a>	PUT /storekeeper/api/items/(int:id), <a href="#">25</a>
GET /storekeeper/api/users, <a href="#">43</a>	PUT /storekeeper/api/items/(int:item_id)/barcodes, <a href="#">29</a>
GET /storekeeper/api/users/(int:id), <a href="#">44</a>	PUT /storekeeper/api/items/(int:item_id)/barcodes/(int:id), <a href="#">30</a>
GET /storekeeper/api/users/(int:id)/config, <a href="#">46</a>	PUT /storekeeper/api/stocktakings/(int:id), <a href="#">34</a>
GET /storekeeper/api/users/(int:id)/config/(string:name), <a href="#">47</a>	PUT /storekeeper/api/stocktakings/(int:id)/close, <a href="#">40</a>
GET /storekeeper/api/vendors, <a href="#">49</a>	PUT /storekeeper/api/stocktakings/(int:id)/items, <a href="#">38</a>
GET /storekeeper/api/vendors/(int:id), <a href="#">50</a>	

---

```
PUT /storekeeper/api/units/(int:id),
    42
PUT /storekeeper/api/users/(int:id),
    45
PUT /storekeeper/api/users/(int:id)/config/(string:name),
    47
PUT /storekeeper/api/vendors/(int:id),
    50
PUT /storekeeper/api/works/(int:id),
    54
PUT /storekeeper/api/works/(int:id)/close-outbound,
    60
PUT /storekeeper/api/works/(int:id)/close-returned,
    61
PUT /storekeeper/api/works/(int:id)/items/(int:item_id),
    58
DELETE /storekeeper/api/acquisitions/(int:id),
    13
DELETE /storekeeper/api/acquisitions/(int:id)/items/(int:item_id),
    18
DELETE /storekeeper/api/customers/(int:id),
    21
DELETE /storekeeper/api/items/(int:id),
    26
DELETE /storekeeper/api/items/(int:item_id)/barcodes/(int:id),
    29
DELETE /storekeeper/api/session, 32
DELETE /storekeeper/api/stocktakings/(int:id),
    35
DELETE /storekeeper/api/stocktakings/(int:id)/items/(int:item_id),
    39
DELETE /storekeeper/api/units/(int:id),
    42
DELETE /storekeeper/api/users/(int:id),
    45
DELETE /storekeeper/api/users/(int:id)/config/(string:name),
    48
DELETE /storekeeper/api/vendors/(int:id),
    50
DELETE /storekeeper/api/works/(int:id),
    55
DELETE /storekeeper/api/works/(int:id)/items/(int:item_id),
    59
```



## A

admin, [10](#)

api, [11](#)

## C

client, [9](#)

## D

dependencies, [9](#)

## F

Flask-Admin, [10](#)

## P

package.sh, [9](#)

## R

RCP API, [11](#)

## S

server, [9](#)

sql, [61](#)