Stock-control and sales system API documentation

API endpoints for a Back-end system of a Stock-control and sales system.

The global variable {{host}} is set to be http://localhost/3000/

Utility

Endpoints for the primary population of the database and for search functionalities that should be implemented in a search bar.

POST /setup

{{host}}setup

Populates the initial data if items from noroff API aren't stored in the db.

POST /search

{{host}}search

Endpoint available to any type of user, allows for search of items or categories, based on criteria such as partial Items names, specific SKU, specific Category or a combination of partial items names and category specific name.

Body raw (json)

json

```json
{
    "Items": "la"
}
```

Signup / Login

Endpoints for signup and login of users.

Duplicate usernames are not allowed, a maximum of 4 duplicate emails are allowed (duplicate emails give relevant discounts upon checkout).
Credentilas are stored in the databse (passwords are hashed+salted and set as BLOB on the db for extra security)

POST /signup

{{host}}signup

Body raw (json)

json

```json
{
    "FirstName": "Ciccio",
    "LastName": "Pasticcio",
    "Username": "chitemmuort",
    "Email": "test@test.com",
    "Password": "test123"
}
```

POST /login

{{host}}login

Logins will provide the user or the Admin with a token that expires after 2 h.

The token needs to be used in the headers to be able to access most parts of the endpoints.

Body raw (json)

json

```json
{
    "Username": "test",
    "Password": "test123"
}
```

Categories

Categories

Endpoints that allows all type of users to see a list of available categories and allows only the Admin to add/update or delete a category.

GET /categories

{{host}}categories

EVeryone can access this endpoint and see a list of all categories.

POST /category

{{host}}category

A category can be added by the Admin only if its name doesn't exist already.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "Name": "Basketball"
}
```

PUT /category/:id

{{host}}category/10

A category can be updated by the Admin to a new name only if such name isn't already in use.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "Name": "Football"
}
```

DELETE /category/:id

{{host}}category/9

A category can be deleted by the Admin only if itsn't in use in any of the available items.

AUTHORIZATION Bearer Token

Token	<token>

Items

Endpoints that allows all type of users to see a list of available items and allows only the Admin to add/update or delete an item.

GET /items

{{host}}items

As requested, guest users can only see in-stock items while registered users can see them all.

POST /item

{{host}}item

An item can be added by the Admin only if name and SKU does not exist, if SKU is in correct format, and if all attributes, apart from Image are in the request body.

AUTHORIZATION Bearer Token

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "Name": "Ball",
    "Price": 10,
    "SKU": "BT456",
    "Quantity": 10,
    "Image": "",
    "Category": 7
}
```

PUT /item/:id

{{host}}item/162

Admin can update whatever attribute of the item by adding it in the request body, could for example be only 1 or could be 3, and this endpoint will update the specific item that has been given as parameter :id with the specific attributes that were sent in the body.

Name can't be in use in other items.

SKU can't be in use in other items and has to be of valid format.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "Name": "Skateboard",
    "SKU": "TC786",
    "Quantity": 50
}
```

DELETE /item/:id

{{host}}item/161

The Admin can Delete an item by giving its id as :id parameter, if the item exists, it will be deleted.

AUTHORIZATION Bearer Token

Token	<token>

Carts

Endpoints that allows users to see their specific cart and the cart items within it or completely empty their cart of all cart items.

Only the admin can see a list of each users cart and the cart items in them.

GET /cart

{{host}}cart

Registered users can see their own cart and the items in it.

AUTHORIZATION Bearer Token

Token	<token>

GET /allcarts

{{host}}allcarts

Admins only can access this endpoint and see all users carts and their cart items.

AUTHORIZATION Bearer Token

Token	<token>

DELETE /cart/:id

{{host}}cart/2

A cart can be emptied by its user, if a wrong cart id is given, an error message is sent.

AUTHORIZATION Bearer Token

Token	<token>

Cart Items

Endpoints that allow users to add items to their cart, update the quantity of a specific cart item, delete/remove a specific cart item from their cart.

POST /cart_item

{{host}}cart_item

An item can be added to a cart as cart item if it isn't already in the cart (in which case they should use the update endpoint) or if it is available amongst all items.

The cart item can be added by sending a request body with either id or Name of the specific item.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "id": 129
}
```

PUT /cart_item/:id

{{host}}cart_item/130

An item id is sent as parameter :id and the endpoint allows a user to update a cart items' quantity, if that item is available in the cart and if the stock quantity of such item is enough to meet the desired quantity.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "Quantity": 5
}
```

DELETE /cart_item/:id

{{host}}cart_item/129

Allows a user to deleete a specific cart item from the cart, if the item id parameter sent matches with any item in the users cart.

AUTHORIZATION Bearer Token

Token	<token>

Orders

Users are allowed to see only their completed orders (as per requirement) and checkout one cartitem at the time from their carts. (customer has specified that to check them all out a loop can be made on the frontend).

Only admins can access the endpoint showing all orders (with all statuses) and the endpoint to update a specific orders' status.

GET /orders

{{host}}orders

Users will see a list of all their completed orders and the relative items within them, while admins will see a list of all users orders (no items specs for each order).

AUTHORIZATION Bearer Token

Token	<token>

GET /allorders

```
{{host}}allorders
```

Only Admins can access this endpoint and will be able to see a list of all users orders and all their related orderitems

AUTHORIZATION Bearer Token

Token	<token>

POST /order/:id

```
{{host}}order/145
```

With this requested endpoint, if an item id that is set as parameter :id matches a cartitem, an in-progress order is created (if it doesn't exist) and consequentially an orderitem too is created. The cart item is removed from the cart and (if there are no other cart items) a checkout message is sent with relevant order id, discount percentage and final cost of the whole order.

Else, a message will tell the customer that the orderitem is added to the order.

Availability checks and adjustments of the stock quanitites are also done within this endpoint.

AUTHORIZATION Bearer Token

Token	<token>

PUT order/:id

```
{{host}}order/3
```

An admin can update an order status from this endpoint.
If complete, the order can ba viewed by the customer in the previous mentioned get orders endpoint.

If cancelled, the endpoint adjusts the quantity of stock that was removed during checkout procedures through POST/order/:id (or POST/cart/checkout)

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "Status": "complete"
}
```

A better checkout

An endpoint that automatically creates an order and orderitems out of all the cart items available in the cart.

It is a better and more reliable solution than the POST/order/:id as it doesn't need extra complexity of having only one "in-progress" order available per each user, and doesn't require too many unneded extra steps for the same result to be achieved.

Also, It accomplishes what earlier is stated by using the POST/order/:id, just so that this extra checkout endpoint that I added can better please and meet the requirements the "customer" had for this project.

Please check my readme for further descriptions on it all works.

POST /cart/checkout

{{host}}cart/checkout

AUTHORIZATION Bearer Token

Token	<token>