# INDEX

[SETUP](#qawb5wfyqnwz)

[CONFIG](#1icgyrs2z74x)

[AUTHENTICATION](#_lomeqcfuec05)

[FYI](#2o9jur6fks0)

[API](#_uro4h6i1or2o)

[STORES](#_6tanfqquyyzs)

[PRODUCTS](#_3gfy3rxohyzb)

[MEMBERS](#_52dygm72e91g)

[ORDERS](#_q5et7cotgyzm)

[ORDER ITEMS](#_7ccpj6xmjqml)

# SETUP

### USING REQUESTS

# We're using the [requests library](http://docs.python-requests.org/en/master/) to make requests. Standard curl will work just as well.

**$ virtualenv env**

**$ source env/bin/activate**

**$ pip install requests**

**$ python**

**>> import requests**

**>> import json**

**>> <REQUEST COMMAND>**

### USING REQUESTS CHROME EXTENSION

An easy alternative is a chrome extension to handle requests, such as [Advanced Rest Client.](https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddffdnphfgcellkdfbfbjeloo)

* Example Advanced Rest Client requests: [ARC EXPORT](https://drive.google.com/open?id=0B6I_PlZVoOpXbzVMWUZNdmF1WFU)

# CONFIG

### DEVELOPMENT

# Switch the port as needed.

BASE API URL: http://127.0.0.1:5000

### PRODUCTION

BASE API URL: https://www.totestore.com

# AUTHENTICATION

Most API endpoints require token authentication.

**Creating an account:**

The easiest way to create an account right now is to purchase something. Alternatively call the Members Create endpoint with an existing auth token.

**Getting your account token:**

To get your token make a request against `/auth` with your member data:

```

res = requests.post(

"https://www.totestore.com/auth",

data={

"email": "<EMAIL>",

"password":"<PASSWORD>"

})

)

print res.json()

```

Auth tokens are currently valid for 30 days. Which means they can be reused between requests. *Don't share your tokens.*

**Using your Auth Token:**

In every request add your auth token to the headers sent.

**headers** = {

"Authentication-Token": "<YOUR\_AUTHENTICATION\_TOKEN>",

"content-type": "application/json"

}

# FYI

* This API is based off [jsonapi specs](http://jsonapi.org/format/).
  + Attributes → Attributes specific to the subject model.
    - ex. if making requests against `/stores` then `name` is an attribute.
  + Relationships → Relationships to other models. Potentially with attributes of foreign models.
    - ex. If making requests against `/products` then `store` is a relationship associated with `products` and `store\_id` is a relationship attribute.
* We loosely follow [Shopify API conventions](https://help.shopify.com/api/reference)
* Most models below have additional optional attributes that can be assigned on creation or update. For a full list, look at the latest specs in the [models](https://github.com/cgil/trie/tree/master/trie/models).
  + We also format requests and responses using these [request schemas.](https://github.com/cgil/trie/tree/master/trie/schemas)
* To view a json response, call json.loads on the request response text.
  + ex. json.loads(requests.get('https://www.totestore.com/stores/').text)

API

# 

# STORES

IN-APP STORES

GET

requests.get('https://www.totestore.com/stores/<STORE\_ID>')

**Filter returned products:**

requests.get(

'https://www.totestore.com/stores/<STORE\_ID>?filter[product]=<PRODUCT1\_ID>,<PRODUCT2\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

GET ALL

requests.get(

'https://www.totestore.com/stores/',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

CREATE

requests.post(

'https://www.totestore.com/stores/',

data=json.dumps({

'data': {

'attributes': {

'name': 'Friendly Store'

},

'type': 'stores'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

PATCH

requests.patch(

'https://www.totestore.com/stores/<STORE\_ID>',

data=json.dumps({

'data': {

'attributes': {

'domain': 'https://amazon.com'

},

'type': 'stores',

'id': '<STORE\_ID>'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

DELETE

requests.delete(

'https://www.totestore.com/stores/<STORE\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

# PRODUCTS

ITEMS IN A STORE

GET

requests.get(

'https://www.totestore.com/products/<PRODUCT\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

GET ALL

requests.get(

'https://www.totestore.com/products/',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

CREATE

requests.post(

'https://www.totestore.com/products/',

data=json.dumps({

'data': {

'attributes': {

'title': 'Jade Harmony Yoga Mat',

'description': 'Combination of grip and grit',

'price': 70.00,

'image': 'http://johnlewis.scene7.com/is/image/JohnLewis/235138385?$prod\_main$'

},

'relationships': {

'store': {

'data': {'type': 'stores', 'id': '<STORE\_ID>'}

}

},

'type': 'products'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

PATCH

**# Here we update both an attribute and a relationship. Both are not required.**

requests.patch(

'https://www.totestore.com/products/<PRODUCT\_ID>',

data=json.dumps({

'data': {

'attributes': {

'image': 'https://image.com/image',

},

'relationships': {

'data': {'type': 'stores', 'id': '<STORE\_ID>''}

},

'type': 'products',

'id': '<PRODUCT\_ID>'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

DELETE

requests.delete(

'https://www.totestore.com/products/<PRODUCT\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

# MEMBERS

A MEMBER OF THE TOTE PLATFORM

GET

requests.get(

'https://www.totestore.com/members/<MEMBER\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

GET ALL

requests.get(

'https://www.totestore.com/members/',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

CREATE

requests.post(

'https://www.totestore.com/members/',

data=json.dumps({

'data': {

'attributes': {

'email': 'customer-email@totestore.com',

'password': 'bwf7675ybvctg456ghvfc',

'stripe\_customer\_id': 'stripe-customer-id'',

},

'type': 'members'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

PATCH

requests.patch(

'https://www.totestore.com/members/<MEMBER\_ID>',

data=json.dumps({

'data': {

'attributes': {

'stripe\_customer\_id': 'new-stripe-customer-id',

},

'type': 'members',

'id': '<MEMBER\_ID>'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

DELETE

requests.delete(

'https://www.totestore.com/members/<MEMBER\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

# ORDERS

AN ORDER FOR ONE OR MORE ITEMS BY A MEMBER FROM A STORE

GET

requests.get(

'https://www.totestore.com/orders/<ORDER\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

GET ALL

requests.get(

'https://www.totestore.com/orders/',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

CREATE

requests.post(

'https://www.totestore.com/orders/',

data=json.dumps({

'data': {

'attributes': {

'financial\_status': 'paid',

'total\_price': 10.00,

'shipping\_address\_city': 'NYC',

'shipping\_address\_country\_code': 'US',

'shipping\_address\_country': 'United States',

'shipping\_address\_1': '401 W 25th St',

'shipping\_address\_zip': '10001',

'shipping\_name': 'Customer Full Name',

},

'relationships': {

'store': {

'data': {'type': 'stores', 'id': <STORE\_ID>}

},

'member': {

'data': {'type': 'members', 'id': <MEMBER\_ID>}

}

},

'type': 'orders'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

PATCH

requests.patch(

'https://www.totestore.com/orders/<ORDER\_ID>',

data=json.dumps({

'data': {

'attributes': {

'fulfillment\_status': 'fulfilled',

},

'type': 'orders',

'id': '<ORDER\_ID>'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

DELETE

**# This does *NOT* delete order items at the moment. Each associated order\_item with this order\_id should also be manually deleted to avoid data inconsistencies.**

requests.delete(

'https://www.totestore.com/orders/<ORDER\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

# ORDER ITEMS

AN INDIVIDUAL PRODUCT IN AN ORDER THAT NEEDS TO BE FULFILLED

GET

requests.get(

'https://www.totestore.com/order\_items/<ORDER\_ITEM\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

GET ALL

requests.get(

'https://www.totestore.com/order\_items/',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

CREATE

requests.post(

'https://www.totestore.com/order\_items/',

data=json.dumps({

'data': {

'attributes': {

'quantity': 3,

},

'relationships': {

'store': {

'data': {'type': 'stores', 'id': <STORE\_ID>}

},

'member': {

'data': {'type': 'members', 'id': <MEMBER\_ID>}

},

'order': {

'data': {'type': 'orders', 'id': <ORDER\_ID>}

},

'product': {

'data': {'type': 'products', 'id': <PRODUCT\_ID>}

}

},

'type': 'order\_items'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

PATCH

requests.patch(

'https://www.totestore.com/order\_items/<ORDER\_ITEM\_ID>',

data=json.dumps({

'data': {

'attributes': {

'quantity': 5,

},

'type': 'order\_items',

'id': '<ORDER\_ITEM\_ID>'

}

}),

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)

DELETE

**# This does *NOT* delete the main order or update the main order total\_price. These need to be done manually at moment.**

requests.delete(

'https://www.totestore.com/order\_items/<ORDER\_ITEM\_ID>',

headers = {

'Authentication-Token': <YOUR\_AUTHENTICATION\_TOKEN>,

'content-type': 'application/json'

}

)