



# SmartSeal

## Smart Seal Inc.

Mark Shekleton  
info@smartseal.io  
860-604-5911

## API Documentation

**v1.1** WORKING COPY

### Digital Identities for Physical Objects

This document is intended for internal and external reference. Please contact [info@smartseal.io](mailto:info@smartseal.io) before sharing. The scope of this document covers an overall system description, API authentication, and supported API methods and endpoints for the dashboard/tag manager, programmer, and end-user use.

By using the SmartSeal API, you agree to the API Terms of Service described in this document.



# Table of Contents

|   |          |
|---|----------|
| <b>Table of Contents</b>  | <b>2</b> |
| <b>1 Smart Seal APIs Terms of Service</b>   | <b>6</b> |
| Section 1: Account and Registration   | 6        |
| a. Accepting the Terms  | 6        |
| b. Entity Level Acceptance  | 6        |
| c. Registration   | 6        |
| d. Subsidiaries and Affiliates  | 7        |
| Section 2: Using Our APIs   | 7        |
| a. Your End Users   | 7        |
| b. Compliance with Law, Third Party Rights, and Other Smart Seal Terms of Service | 7        |
| c. Permitted Access   | 7        |
| d. API Limitations  | 7        |
| e. Open Source Software   | 8        |
| f. Communication with Smart Seal  | 8        |
| g. Feedback   | 8        |
| h. Non-Exclusivity  | 8        |
| Section 3: Your API Clients   | 8        |
| a. API Clients and Monitoring   | 8        |
| b. Security   | 8        |
| c. Ownership  | 9        |
| d. User Privacy and API Clients   | 9        |
| Section 4: Prohibitions and Confidentiality                                       | 9        |
| a. API Prohibitions   | 9        |
| b. Confidential Matters   | 10       |
| Section 5: Content  | 10       |
| a. Content Accessible Through our APIs  | 10       |
| b. Submission of Content  | 10       |
| c. Retrieval of content   | 11       |
| d. Data Portability   | 11       |
| e. Prohibitions on Content  | 11       |
| Section 6: Brand Features; Attribution  | 11       |
| a. Brand Features   | 11       |



|   |           |
|---|-----------|
| b. Attribution                              | 12        |
| c. Publicity                                | 12        |
| d. Promotional and Marketing Use            | 12        |
| Section 7: Privacy and Copyright Protection | 12        |
| a. Smart Seal Privacy Policies              | 12        |
| b. Smart Seal DMCA Policy                   | 12        |
| Section 8: Termination                      | 13        |
| a. Termination                              | 13        |
| b. Your Obligations Post-Termination        | 13        |
| c. Surviving Provisions                     | 13        |
| Section 9: Liability for our APIs           | 13        |
| a. WARRANTIES                               | 13        |
| b. LIMITATION OF LIABILITY                  | 14        |
| c. Indemnification                          | 14        |
| Section 10: General Provisions              | 14        |
| a. Modification                             | 14        |
| b. U.S. Federal Agency Entities             | 15        |
| c. General Legal Terms                      | 15        |
| <b>2 System Description</b>                 | <b>16</b> |
| 2.1 Passthrough Mode                        | 16        |
| 2.2 Call Mode                               | 16        |
| 2.3 A Note on Tag Types                     | 16        |
| <b>3 API Authorization</b>                  |           |
| <b>Management and Tokens</b>                | <b>18</b> |
| 3.1 Through the rest_auth package:          | 18        |
| 3.1.1 Login                                 | 18        |
| 3.1.2 Logout                                | 18        |
| 3.1.3 Password Reset                        | 18        |
| 3.1.4 Password Reset Confirmation           | 19        |
| 3.1.5 Password Change                       | 19        |
| 3.1.6 User Details                          | 19        |
| 3.2 Through the rest_framework package:     | 20        |
| Login                                       | 20        |
| Logout                                      | 20        |
| 3.3 Through the drf_auth package:           | 20        |
| 3.3.1 API Root                              | 20        |
| Users                                       | 20        |



|                                      |           |
|--------------------------------------|-----------|
| User List                            | 20        |
| Activation                           | 21        |
| Me                                   | 21        |
| Resend Activation                    | 21        |
| Reset Username                       | 22        |
| Set Password                         | 22        |
| Set Username                         | 22        |
| Token Create                         | 22        |
| Token Destroy                        | 22        |
| 3.4 Token Authentication             | 23        |
| 3.4.1 Python Example                 | 23        |
| 3.5 Permission Levels                | 23        |
| <b>4 General Use API</b>             | <b>24</b> |
| 4.1 Tags                             | 24        |
| 4.1.1 Data Definition                | 24        |
| 4.1.2 Get All Tags                   | 25        |
| 4.1.3 Create a New Tag               | 25        |
| 4.1.4 Get Tag Detail                 | 25        |
| 4.1.5 Update a Tag                   | 26        |
| 4.1.6 Get Individual Tag Attributes  | 26        |
| 4.1.7 Delete a Tag                   | 26        |
| 4.2 Scans                            | 26        |
| 4.2.1 Data Definition                | 26        |
| 4.2.2 Get All Scans                  | 28        |
| 4.2.3 Get Scan Detail                | 28        |
| 4.2.4 Get Individual Scan Attributes | 28        |
| <b>5 Tag Authentication API</b>      | <b>28</b> |
| <b>6 Dashboard API</b>               | <b>29</b> |
| 6.1 Heatmap                          | 29        |
| 6.2 Hourly Scans                     | 29        |
| 6.3 Highest Number of Scans          | 30        |
| 6.4 Tag Authentication Totals        | 30        |
| 6.5 High Failure Tags                | 30        |
| <b>7 Tag Manager API</b>             | <b>30</b> |
| <b>8 Billing API</b>                 | <b>31</b> |





# 1 Smart Seal APIs Terms of Service

Thank you for using Smart Seal's APIs, other developer services, and associated software (collectively, "APIs"). By accessing or using our APIs, you are agreeing to the terms below. If there is a conflict between these terms and additional terms applicable to a given API, the additional terms will control for that conflict. Collectively, we refer to the terms below, any additional terms, terms within the accompanying API documentation, and any applicable policies and guidelines as the "Terms." You agree to comply with the Terms and that the Terms control your relationship with us. So please read all the Terms carefully. If you use the APIs as an interface to, or in conjunction with other Smart Seal products or services, then the terms for those other products or services also apply.

Under the Terms, "Smart Seal" means Smart Seal Inc., with offices at 280 Phoenix Street, Vernon, Connecticut 06066, United States, unless set forth otherwise in additional terms applicable for a given API. We may refer to "Smart Seal" as "we", "our", or "us" in the Terms.



## 2 System Description

Depending on your setup, there are two modes that the SmartSeal Authentication System operates in, passthrough mode or call mode. These modes are outlined in sections 2.1 and 2.2.

### 2.1 Passthrough Mode

In passthrough mode, the URL on the NFC tag points directly to the Smart Seal server. For example:

`https://<your_domain>.smartseal.io/76AFF09028BC3837DE9407FA8D2C47264BBC837876AF09028BC3837DE9407FA8D2C47264BBC8378` sends the request from the customer's phone browser directly to our server. We collect all the necessary information from each scan, perform the authentication, and redirect to your secure authentication page.

### 2.2 Call Mode

In call mode, the URL on the NFC tag points directly to the client's server. For example:

`https://<your_domain>.com/authenticate/76AFF09028BC3837DE9407FA8D2C47264BBC837876AFF09028BC3837DE9407FA8D2C47264BBC8378` sends the request from the customer's phone browser directly to your server. To render the page correctly, scripts on this page collect the scan data (useragent, location, etc) and the encrypted payload and send it to the Smart Seal server through a POST request. The return message will contain the authentication status and other useful information about the tag (unique identifier, etc..). This information is then used to properly render the page.

### 2.3 A Note on Tag Types

There are two main types of tags. Regular tags and tag-tamper variants. For many applications, tag-tamper variants are not used. However, the Smart Seal system handles both types. If you are not using a tag-tamper variant, it is safe to ignore tag tamper related fields. "tt\_type" is set to "true" if this is a tag-tamper variant. "tt\_status" is set to "false" when the tamper loop is broken. On regular tags, this flag does nothing.



## 3 API Authorization Management and Tokens

### 3.1 Through the rest\_auth package:

#### 3.1.1 Login

```
POST https://<my_domain>.smartseal.io/rest-auth/login/
```

Check the credentials and return the REST Token if the credentials are valid and authenticated.

Calls Django Auth login method to register User ID in Django session framework

Accept the following POST parameters: username, password  
Return the REST Framework Token Object's key.

#### 3.1.2 Logout

```
GET, POST, HEAD https://<my_domain>.smartseal.io/rest-auth/logout/
```

Calls Django logout method and delete the Token object assigned to the current User object.

Accepts/Returns nothing.

#### 3.1.3 Password Reset

```
POST https://<my_domain>.smartseal.io/rest-auth/password/reset/
```

Accepts the following POST parameters: email  
Returns the success/fail message.





### 3.1.4 Password Reset Confirmation

```
POST https://<my_domain>.smartseal.io/rest-auth/password/reset/confirm/
```

Accepts the following POST parameters: token, uid,  
new\_password1, new\_password2  
Returns the success/fail message.

### 3.1.5 Password Change

```
POST https://<my_domain>.smartseal.io/rest-auth/password/change/
```

Calls Django Auth SetPasswordForm save method.

```
{
  "new_password1": "",
  "new_password2": ""
}
```

Accepts the following POST parameters: new\_password1, new\_password2  
Returns the success/fail message.

### 3.1.6 User Details

```
GET, PUT, PATCH https://<my_domain>.smartseal.io/rest-auth/user/
```

Reads and updates UserModel fields  
Accepts GET, PUT, PATCH methods.

Default accepted fields: username, first\_name, last\_name  
Default display fields: pk, username, email, first\_name, last\_name  
Read-only fields: pk, email

Returns UserModel fields.



## 3.2 Through the rest\_framework package:

### Login

api-auth/ ^login/\$ [name='login']  
[https://<my\\_domain>.smartseal.io/api-auth/login/](https://<my_domain>.smartseal.io/api-auth/login/)

### Logout

api-auth/ ^logout/\$ [name='logout']  
[https://<my\\_domain>.smartseal.io/api-auth/logout/](https://<my_domain>.smartseal.io/api-auth/logout/)

## 3.3 Through the drf\_auth package:

<https://pypi.org/project/django-drf-auth/>

### 3.3.1 API Root

The default basic root view for DefaultRouter

auth/ ^\$ [name='api-root']  
auth/ ^\.(?P<format>[a-z0-9]+)/?\$ [name='api-root']

```
GET https://<my_domain>.smartseal.io/auth/
```

### Users

auth/ ^users/(?P<pk>[^\.]+)/\$ [name='user-detail']  
auth/ ^users/(?P<pk>[^\.]+)\.(?P<format>[a-z0-9]+)/?\$ [name='user-detail']

### User List

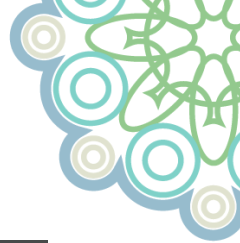
auth/ ^users/\$ [name='user-list']  
auth/ ^users\.(?P<format>[a-z0-9]+)/?\$ [name='user-list']

```
GET, POST, HEAD https://<my_domain>.smartseal.io/auth/users/
```

```
GET, POST, HEAD https://<my_domain>.smartseal.io/auth/users/<?>
```

POST:

```
{
```



```
"email": "",
"username": "",
"password": ""
}
```

### Activation

```
POST https://<my_domain>.smartseal.io/auth/users/activation/
```

POST

```
{
  "uid": "",
  "token": ""
}
```

### Me

```
GET, PUT, PATCH, DELETE https://<my_domain>.smartseal.io/auth/users/me/
```

PUT, PATCH:

```
{
  "email": "mark@smartseal.io",
  "id": 1,
  "username": "admin"
}
```

### Resend Activation

```
https://<my_domain>.smartseal.io/auth/users/resend_activation/
```

```
https://<my_domain>.smartseal.io/auth/users/reset_password/
```

```
https://<my_domain>.smartseal.io/auth/users/reset_password_confirm/
```



### Reset Username

```
https://<my_domain>.smartseal.io/auth/ users/reset_username/
```

```
https://<my_domain>.smartseal.io/auth/users/reset_username_confirm/
```

### Set Password

```
https://<my_domain>.smartseal.io/auth/ users/set_password/
```

### Set Username

```
https://<my_domain>.smartseal.io/auth/users/set_username/
```

### Token Create

Use this method to obtain a user authentication token.

```
POST https://<my_domain>.smartseal.io/auth/token/login/
```

POST:

```
{
  "password": "",
  "username": ""
}
```

RETURN:

```
{
  "auth_token": "e109dea521ec9b8ff061fd3471548993e6cba040"
}
```

### Token Destroy

Use this endpoint to logout user (remove user authentication token)

```
POST https://<my_domain>.smartseal.io/auth/token/logout
```



## 3.4 Token Authentication

### 3.4.1 Python Example

```
import json
import requests
import getpass

def login():
    username = raw_input("Username: ")
    password = getpass.getpass(prompt="Password: ", stream=None)
    credentials = {"username": str(username), "password": str(password)}
    response = requests.post("https://<my_domain>.smartseal.io/rest-auth/login/",
data=credentials)
    #if you forget the last /, you will have problems

    if response.status_code is not 200:
        print("ERROR: Status Code " + str(response.status_code))
        json_data = response.json()
        for i in json_data:
            print(i.upper().encode('ascii', 'ignore') + ": " + str(json_data[i])[3:][-2])
            raise Exception("PROGRAM ABORTED: Login error")
    else:
        json_data = response.json()
        print("Token Received")
        return json_data.get("key")

def upload_tag_data(token, tag_data):
    headers = {"Authorization": "Token " + token, 'Content-Type': 'application/json'}
    tag_data_json = json.dumps(tag_data, default=str)
    response = requests.post("https://<my_domain>.smartseal.io/api/tag_list/",
headers=headers, data=tag_data_json)

    if response.status_code is not 201:
        print("ERROR: Status Code " + str(response.status_code))
        json_data = response.json()
        for i in json_data:
            print(i.upper().encode('ascii', 'ignore') + ": " +
str(json_data[i]))
    else:
        print("Tag successfully loaded into SmartCloud database.")
```

## 3.5 Permission Levels

User groups:



Super Admin: Full access. Able to modify any record or user.

Admin: Can add users and has limited permissions to modify records. Cannot delete tags or modify keys.

Programmer: Only has write access for programming tags.

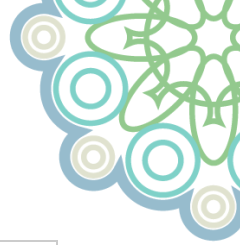
User: Mostly read-only access with limited ability to modify tag data through dashboard.

## 4 Customer Data API

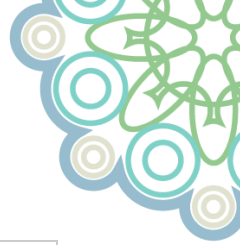
### 4.1 Tags

#### 4.1.1 Data Definition

| Attribute            | Type                      | Example          | Explanation  |
|----------------------|---------------------------|------------------|--|
| uid                  | 7 byte hex string         | '123456789abcde' | Universal Identifier from tag  |
| programmer_id        | character string, 127 max |                  | ID of programmer user  |
| programmed_timestamp | UTC date and time         |                  | Datestamp when tag was programmed  |
| tag_valid            | true/false                |                  | True: Tag will authenticated and will be counted as an active, billable tag<br>False: Tag will not authenticated and will not be counted as an active, billable tag<br>Use this flag to deactivate tag instead of deleting it. |
| tag_expired          | true/false                |                  | True: Tag is expired and is no long billable<br>False: Tag has not expired. May or may not be billable, depending on tag_valid flag  |
| tt_type              | true/false                |                  | True: Tag-tamper variant<br>False: Non-tag-tamper variant  |
| tt_status            | true/false                |                  | True: Tamper loop is intact<br>False: Tamper loop is broken  |



|                     |                   |  |  |
|---------------------|-------------------|--|--|
| greatest_auth_count | integer           |  | Highest tag scan count that has been authenticated   |
| count_limit         | integer           |  | Null: Unlimited scans<br>Integer: Tag will become invalid and non-billable after this many scans   |
| series              | character string  |  | Product series ID  |
| batch_id            | character string  |  | Product batch ID   |
| published           | true/false        |  | True: Tag is published, active, and billable unless tag_valid flag is false<br>False: Tag is not yet published and is not active and billable. Will become active when published_timestamp date is met |
| published_timestamp | UTC date and time |  | Date when tag becomes active and billable.   |
| coa_id              | character string  |  | Certificate of authenticity ID, DEPRECATED   |
| customer_id         | character string  |  | Customer ID  |
| tag_expiration      | UTC date and time |  | Expiration date. After expiration date, tag will become inactive and non-billable  |
| product_expiration  | character string  |  | Optional product expiration date if product is perishable  |
| redirect_url        | URL string        |  | URL for passthrough mode of non-tag-tamper variant tags  |
| redirect_ttopen     | URL string        |  | URL for passthrough mode of tag-tamper variant tags when tamper loop is broken   |
| redirect_ttclose    | URL string        |  | URL for passthrough mode of tag-tamper variant tags when tamper loop is intact   |
| last_scan_date      | UTC date and time |  | Last time this tag was scanned   |
| comments            | character string  |  | Comments on tag or product for future reference  |



|                       |                  |                                       |   |
|-----------------------|------------------|---------------------------------------|---|
| fail_counter          | integer          |                                       | Number of times tag has failed authentication                               |
| expired_fail_counter  | integer          |                                       | Number of times tag has failed authentication because tag expired           |
| pass_counter_sealed   | integer          |                                       | Number of times tag has passed authentication and the tamper loop is intact |
| pass_counter_unsealed | integer          |                                       | Number of times tag has passed authentication and the tamper loop is broken |
| description           | character string | "Bahama Design Cotton T-shirt, Blue." | Description of product  |

### 4.1.2 Get All Tags

```
GET https://<my_domain>.smartseal.io/api/tags
```

Returns a list of all the tags and their attributes.

### 4.1.3 Create a New Tag

```
POST https://<my_domain>.smartseal.io/api/tags
```

Must send all attributes.

### 4.1.4 Get Tag Detail

```
GET https://<my_domain>.smartseal.io/api/tags/<uid>
```

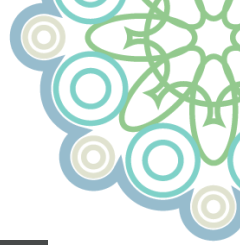
Returns a list of all the attributes of a single tag.

### 4.1.5 Update a Tag

```
PUT https://<my_domain>.smartseal.io/api/tags/<uid>
```

Must send all attributes.





```
PATCH https://<my_domain>.smartseal.io/api/tags/<uid>
```

Only need to send attributes that need to be updated.

### 4.1.6 Get Individual Tag Attributes

```
GET https://<my_domain>.smartseal.io/api/tags/<uid>/<attribute>
```

Returns the value of the attribute for the given UID.

### 4.1.7 Delete a Tag

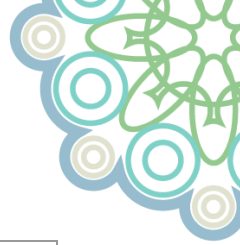
```
DELETE https://<my_domain>.smartseal.io/api/tags/<uid>
```

Not Recommended. Billing is only for activated tags. To pause billing on a tag, update tag\_valid to false.

## 4.2 Scans

### 4.2.1 Data Definition

| Attribute    | Type                           | Example                | Explanation   |
|--------------|--------------------------------|------------------------|---|
| uid          | 7 byte hex string              | '123456789ABCDE'       | Universal Identifier from tag                               |
| scan_date    | UTC time and date              | '2020-05-18T19:53:02Z' | Date and time of scan                                       |
| tt_status    | true/false                     |                        | Indication of tamper loop status. Ignore if non-tt variant. |
| useragent    | text string                    |                        | Useragent data from scan                                    |
| ipv4_address | Character string, max 14 chars |                        | IP address  |
| ip_address   | Character string, max 45 chars |                        | IP address  |
| ip_location  | Character                      |                        | IP address based location                                   |



|           |                       |  |                       |
|-----------|-----------------------|--|-----------------------|
|           | string, max 200 chars |  |                       |
| auth_stat | integer               |  | Authorization status: |
| count     | integer               |  | Tag scan count        |
| lat       | DDD.ddddddd           |  | Geolocation for scan  |
| lng       | DDD.ddddddd           |  | Geolocation for scan  |

## 4.2.2 Get All Scans

```
GET https://<my_domain>.smartseal.io/api/scans
```

## 4.2.3 Get Scan Detail

```
GET https://<my_domain>.smartseal.io/api/scans/<scan_number>
```

## 4.2.4 Get Individual Scan Attributes

```
GET https://<my_domain>.smartseal.io/api/scans/<scan_number>/<attribute>
```

# 5 Tag Authentication API

<<< UNDER CONSTRUCTION >>>

POST https://<my\_domain>.smartseal.io/api/authenticate\_tag/<url\_payload>

POST:

- url\_payload,
- lat,
- lng,
- useragent info,
- IP address

RETURNS:

404 BAD REQUEST, OR:

- authstat
- tag info
- scan data info

<< Insert JS code here >>



[https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref\\_nav\\_all](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_nav_all)

## 6 Dashboard API

Dashboard GET requests are intended to be used with AJAX and are authorized through session authentication. They require the CSRF token to be included in the header, as in the following example:

<https://www.geeksforgeeks.org/handling-ajax-request-in-django/>

Note: These endpoints are also available through token authentication

### 6.1 Heatmap

```
GET https://<my_domain>.smartseal.io/api/heatmap/<timeframe>/<authstat>
```

Where:

timeframe = "pastday", "pastweek", "pastmonth", "pastyear", or "alltime"  
authstat = "no\_auth", "pass\_auth", "pass\_tt\_closed", "pass\_tt\_open",  
"fail\_not\_published", "fail\_tag\_expired", "fail\_tag\_invalid", "fail\_count\_expired",  
"fail\_bad\_cmac", "all\_pass", "all\_fail", or "all"

Returns:

Lat: <signed decimal, 9 digits, 6 decimal places>

Ing: <signed decimal, 9 digits, 6 decimal places>

### 6.2 Hourly Scans

```
GET https://<my_domain>.smartseal.io/api/scans_per_hour/<hours>/<authstat>/
```

Where:

hours = 1 through +inf, or "all"

authstat = "no\_auth", "pass\_auth", "pass\_tt\_closed", "pass\_tt\_open", "fail\_not\_published",  
"fail\_tag\_expired", "fail\_tag\_invalid", "fail\_count\_expired", "fail\_bad\_cmac", "all\_pass", "all\_fail",  
or "all"

Returns:

hour: <1: past hour, 2: between 1 and two hours ago, 3: between 2 and 3 hours ago, etc..>

scan\_count: <number of scans that happened at this hour>



## 6.3 Highest Number of Scans

```
GET https://<my_domain>.smartseal.io/api/highest_scans
```

## 6.4 Tag Authentication Totals

```
GET https://<my_domain>.smartseal.io/api/auth_totals
```

Returns:

```
no_auth:
pass_auth:
pass_tt_closed:
pass_tt_open:
fail_not_published:
fail_tag_expired:
fail_tag_invalid:
fail_count_expired:
fail_bad_cmac:
all_pass:
all_fail:
all:
```

## 6.5 High Failure Tags

```
GET https://<my_domain>.smartseal.io/api/hot_tags
```

# 7 Tag Manager API

<<UNDER CONSTRUCTION>>

ENDPOINT:

PAGINATION ARGUMENTS:

FILTER ARGUMENTS:

MULTIPLE UPDATES:



## 8 Billing API

```
GET https://<my_domain>.smartseal.io/api/billing/
```

Returns: Current billing information, updated daily

## 9 Known Issues

1. Permissions are may be too restrictive
2. Need code examples for AJAX
3. Dashboard login redirects
4. Billing totalizer not complete
5. Data dictionaries not complete



# SmartSeal

### Contact information

Mark Shekleton

[info@smartseal.io](mailto:info@smartseal.io)

860-604-5911