

# API Documentation

# CSSI PERSONAL

# Version .1

|  |  |  |
| --- | --- | --- |
| By: | Chad Clites | [chad@extant.digital](mailto:chad@extant.digital) |
| For: | John Glavin | [jglavin@eyeforsecurity.com](mailto:jglavin@eyeforsecurity.com) |
| For: | James Ettore | jettore@eyeforsecurity.com |

Revisions

|  |  |  |
| --- | --- | --- |
| 09/11/2018 | Public release | Chad Clites |
| 10/29/2018 | Added distinct parameter | Chad Clites |
|  |  |  |

**General Information**

The CSSI PERSONAL API provides the following:

* Aliases
* SSN validation
* Date of Birth
* Date of Death
* Prior Addresses
* Prior Phone numbers
* Relatives

Access to the CSSI API requires an API token, and an account key, both of which can be found on the settings page. Once an account has been created, the site administrators will grant access.

**End Point**

Ordering = POST: https://api.cssidata.com/api/cssi/data

***Input Parameters***

|  |  |
| --- | --- |
| first\_name | required |
| middle\_name | optional |
| last\_name | required |
| address | required – street address |
| city | required |
| state | required |
| zip | required |
| email | optional |
| check\_type | Personal |
| distinct | 8-12 string nonce (optional)\*\* |

\*\* The distinct nonce must be a random and unique string of 8-12 characters and must be unique per each request. This field was created to prevent the accidental submission of duplicate checks. If the field is not included, the API user will be responsible for any charges resulting from generating accidental duplicates.

**SAMPLE CODE**

**\*\*\* TESTING NOTE: This check can be tested with any user data. The default data here will not return any results.**

$api\_key = "<api token>";

$key = “<account key>”**;**

$url = "https://api.cssidata.com/api/cssi/data?token=$api\_key&key=$key";

$params = array(

"first\_name" => "Bruce",

"last\_name" => "Dickinson",

"address" => "220 Arcadia Avenue",

"state" => "MN",

"city" => "London",

"zip" => "59999",

“birthday” => “09/09/2000”,

"email" => "chad@extant.digital",

"check\_type" => "personal"

);

$curl = curl\_init();

curl\_setopt\_array($curl, array(

CURLOPT\_URL => $url,

CURLOPT\_RETURNTRANSFER => true,

CURLOPT\_ENCODING => "",

CURLOPT\_MAXREDIRS => 10,

CURLOPT\_TIMEOUT => 300,

CURLOPT\_CUSTOMREQUEST => 'POST',

CURLOPT\_POSTFIELDS => $params,

));

$response = curl\_exec($curl);

curl\_close($curl);

**Return Data**

message = <request error>

status = -1

tracking = null

message = <system error>

status = 0

tracking = null or <cssi-generated tracking id>

message = <search data>

status = 1,

tracking = <cssi-generated tracking id>

**Return Data Parameters**

### Repeatable fields

The following fields are repeatable and should be accounted for when programming:

* people/person
* people/person/names/name
* people/person/sharingSSNs/sharingSSN
* people/person/akalist/AKA
* people/person/SSNs/SSNInfo
* people/person/DODs/DOD
* people/person/DOBs/DOB
* people/person/addresses/address
* people/person/addresses/phonelist/phone
* people/person/addresses/phones/phone
* people/person/relatives/relative
* people/person/otherPhones/phone
* people/person/emails/email
* people/person/driversLicenses/driversLicense

### Empty Fields

Some tags will always be empty. This is for backwards compatibility for some clients migrating from different endpoints. Those tags are as follows:

* …/address/propertyPhotoURLs
* …/address/propertyPhotosAddress
* …/address/voterRegistrations
* …/address/distance
* …/phone/toDate
* …/phone/fromDate
* …/person/pictureURLs
* …/person/firstDate
* …/person/lastDate

### Notes

Please note that …/address/phonelist and …/address/phones will contain the same data. The data exists in both fields for backwards compatibility. Additionally, .../person/names will contain the same data (in a different format) as …/person/akalist

Data Explanations and Clarifications

This section is intended to describe the meaning behind some of the fields and give some context to the developer to determine whether they would be useful in the client’s application. Each field is listed by its XPath.

**/people/person/sharingSSNs**

Sharing SSNs often appear when more than one individual has been associated with the SSN. This does not necessarily indicate fraud. These are just individuals who have at one point used that SSN – usually by mistake (ex: a typo)

**/people/person/SSNs**

Each individual can have more than one SSN associated with them. When they do, you will see more than one SSNInfo tag. Not all SSNs have SSNState and SSNYears tags populated. This is because not all SSN issuance ranges are known. For more information, please see SSA’s information on [SSN Randomization](https://www.ssa.gov/employer/randomization.html).

**…/address/phonelist and …/address/phones**

These two fields are twins. They will always contain the same data in the same format. They are duplicated for backwards compatibility purposes.

These fields indicate the **current phone number at that address** – NOT the individual’s phone number when they lived there.

**…/relative/DOB**

Relative DOBs will always be in the format of MM/00/YYYY – the day will be restricted and listed as 00.

**/people/person/otherPhones**

The “otherPhones” are phone numbers that are associated with the individual. They are not tied to a specific <address> or <name>

**…/person/DOBs**

The DOBs provided in the response are in order of reliability. There is no reliability score provided in the output, but they are in ranked order by default.