idea v1.0

Postman shared collection

https://www.getpostman.com/collections/edc3eb18512ec586e47a

How to Import Postman collection

Requirments

All our API's requires https (encrypted) connections , do not use http

POST idea

URL: https://api.ipscreener.com/v1/idea

Method: POST

Headers

Required:
authorization=[value]

Body

Required:

Username=[string]

Reference=[string]

Title=[string]

Summary=[string]

Header

authorization : An API authorization key must be sent with all requests. You need to contact the IPscreener support team to get the API authorization key.

Body

Username: What user is making the request. Must be an valid email. If the email adress dosen't already exist, the system will create a new user.

Reference: Reference name for your case/search.

Title: The title of your idea, which is included in the search quary.

Summary: A description of your idea. Recommended length is half a A4-page or more.

(**note**: a request MUST contain a few words or your search won't process.

Success Response:

data: the session ticket value associated with a search request used to GET and PUT the search results.

case: Your case Id used to GET/PUT your search results.

url: A url to your search result inside IPscreener with automatic login.

expire: a ticket is valid for 1 hour before expired and ticket is invalid. Time is displayed in UNIX

Success Response

Body content format: JSON

The below response will be returned once the search request has been processed. To view the result please use the GET idea API with the case value.

HTTPS Success Response 200 OK

```
{
    "status": "success",
    "message": "New idea have been created successfully",
    "data": {
        "case": "20696",
        "url": "https://my.ipscreener.com/token
/3VId5AkOzXTxDFTNgBTDz1MTR8Mw",
        "expire": 1589535977
    }
}
```

Error Responses

The below response will be returned if a requierd field is missing or left blank. **(Username, Reference, Title or Summary.)**

Error response

```
{
    "status": "error",
    "message": "Data in required fields are missing"
}
```

The below response will be returned if the e-mail format is wrong. **E.g "#test@e.mail**

Email format

```
{
    "status": "error",
    "message": "Wrong format of email address"
}
```

The below response will be returned if the e-mail is already in use on another account.

Note

Supported languages: english, german, french, japanese.

Email already registered

```
{
    "status": "error",
    "message": "Username does not exists in your account"
}
```

The below response will be returned if your API key is wrong or invalid.

Error response API

```
{
    "status": "error",
    "message": "API key is not valid"
}
```

The below response will be returned if your query is to short or language is unsupported.

Short query/Language

```
{
    "status": "error",
    "message": "The input was short and our language
detection algorithm not smart enough to understand,
please add some more text."
}
```

Sample Call (cURL)

GET idea

URL: https://api.ipscreener.com/v1/idea

Method: GET

Headers

Headers

authorization=[value]

Body

case=[value]

Success Response Body

data: contains all case data Case=[value]

case: the session ticket value associated with a search request used to PUT the search results.

Success.

Response:

expire: how long a case id is valid. Before expiring. Time is displayed in UNIX

Header

authorization : An API authorization key must be sent with all requests. You need to contact the IPscreener support team to

Success Response:

Body content format: JSON

```
{
    "index": [
        {
            "id": "1",
            "name": "Master"
        },
        {
            "id": "999",
            "name": "Novelty screening"
        },
        {
            "id": "54523",
            "name": "Infringement screening"
        }
    ],
    "result": {
        "1": [],
        "999": [],
        "54523": [
            {
                "position": "1",
                "rating": "Similar",
```

Sample Call (cURL)

```
curl --location --request POST 'https://api.ipscreener.com/v1/idea' \
--header 'key: <value>' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'username=[email]' \
--data-urlencode 'reference=Solid-state drive' \
--data-urlencode 'title=Solid-state drive' \
--data-urlencode 'summary=[text]' \
--data-urlencode 'case=[value]'
```

PUT idea

URL: https://api.ipscreener.com/v1/idea

Method: PUT

Headers

Headers

authorization=[value]

Body

case=[value]

Success Response Body

data: a new URL token is generated.

Response:

expire: how long a case id is valid. Before expiring. Time is displayed in UNIX

Header

authorization : An API authorization key must be sent with all requests. You need to contact the IPscreener support team to

Success Response:

Body content format: JSON

When checking if data is ready for retrieval you only use the parameter ticket value. The response below will be returned once the search request has been processed and is ready for delivery.

Sample Call (cURL)

```
curl --location --request PUT 'https://api.ipscreener.com/v1/idea' \
--header 'key: <value>' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'case=[value]'
```

GET Download

HTTPS Success Response 200 OK

Body content format: JSON

When checking if data is ready for retrieval you only use the parameter ticket value. The response below will be returned once the search request has been processed and is ready for delivery.

This API is used to retrive data from an expired ticket.

```
{
    "status": "success",
    "message": "New token have been created successfully",
    "data": {
        "case": "20718",
        "url": "https://my.ipscreener.com/token/GJhhi9aDxovk",
        "expire": 1589810666
    }
}
```

Error Response

The below response will be returned if your API key is wrong or invalid.

get the API authorization key.

Method: GET

Body

Headers

Case: the session ticket value **required:** associated with a search request authorization=[value] used to GET the search results.

Body

required:

Success Response:

type=[keyword] **index value:** After the parameter automatch-result there is an integer value, e.g. index-1. This value indicates the specific index and associated settings used for performing the matching procedure. If several indexes are targeted with a search (maximum three in parallel) each result list will be presented separately one after the other.

E.g type=pdf_all **position**: The position refers to the record number of the hit in the search results, where the search results are sorted on the relevance score value in descending order.

rating: Rating refers to your ranking made; background, relevant or similar. Noise or unranked documents won't appear.

image: It creates a link to the first image, if one exist. Images file type is .png.

note: Comments made on this document.

document type: C an be two different types

- Applicant, a not yet granted patent.
- Patent, a granted patent.

URL: https://api.ipscreener.com/v1/download

Method: GET

Body

Headers

Case: the session ticket value **required:** associated with a search request authorization=[value] used to GET the search results.

Body

required:

Success Response:

type=[keyword] **index value:** After the parameter automatch-result there is an integer value, e.g. index-1. This value indicates the specific index and associated settings used for performing the matching procedure. If several indexes are targeted with a search (maximum three in parallel) each result list will be presented separately one after the other.

E.g type=pdf_all **position**: The position refers to the record number of the hit in the search results, where the search results are sorted on the relevance score value in descending order.

rating: Rating refers to your ranking made; background, relevant or similar. Noise or unranked documents won't appear.

image: It creates a link to the first image, if one exist. Images file type is .png.

note: Comments made on this document.

document type: C an be two different types

- Applicant, a not yet granted patent.
- Patent, a granted patent.

Header:

authorization: An API authorization key must be used with all requests. You need to contact the IPscreener support team to get the API authorization key.

Body:

case: case id from POST idea.

type: Export options, they are case sensative.

- pdf_ranked (Only ranked document)
- pdf_all (All documents)
- excel_ranked (Only ranked document)
- excel_all (All documents)
- word_ranked (Only ranked document)

Success Response

data: Retrieval of a complete PDF, Excel or Word document encoded in Base64 format.

The Download API is used to retrieve the original patent document(s) corresponding to a search query. From the requested case id, the API request returns a PDF,Excel or Word file in base64 format based on ranking.

Success Response:

The below response will be returned if your case token is;

- wrong/invalid
- try to open a case id that belongs to another customer/company or doesn't exist
- Special char included in id, e.g 207"4
- tab, whitespace or enter is used in case id field, e.g 20 74

Body content format: JSON

HTTPS Success Response 200 OK

```
{
    "status": "success",
    "message": "Report has been successfully fetched",
    "data": "<Complete document encoded in Base64 format>"
}
```

No permission

```
{
    "status": "error",
    "message": "You dont have permission to do that"
}
```

Error Response:

No valid API key

Sample Call (cURL)

```
{
    "status": "error",
    "message": "API key is not valid"
}
```

Sample Call (cURL)

```
curl --location --request PUT 'https://api.ipscreener.com/v1/idea' \
--header 'key: [value]' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'case=20718'
```

Wrong Case id

```
{
    "status": "error",
    "message": "You dont have permission to do that"
}
```

Sample call (cURL)

Sample cURL

```
curl --location --request GET 'https://api.ipscreener.com/v1
/download' \
--header 'key: Auth-key' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'case=20990' \
--data-urlencode 'type=word_ranked'
```


```
{
    "image": "https://my.ipscreener.com/img.php?id=US-10338825-B2-firstpage.png",
    "note": null,
    "document_type": "patent",
    "patent_number": "US10338825",
    "kindcode": "B2",
    "publication_date": "2017-10-05",
    "priority_date": "2015-03-31",
    "title": "Managing SSD wear rate in hybrid storage arrays",
    "abstract": "A technique for use in balancing flash drive wear in data storage systems is disclosed. Multiple sets of flash drives are identified where data is stored as multiple slices striped across the respective sets of flash drives. A write rate at which data will be written to the multiple slices stored on the set of flash drives during a next time interval is predicted. A number of bytes that can be written to each set of flash drives is determined. A metric representative of a wear rate is determined for each set of flash drives. HDD relocation candidates are identified, and a relocation process to relocate identified slices initiated.",
    "claim": "1. A method for use in balancing solid state drives (SSD) wear in data storage systems, the method comprising:\nidentifying multiple sets of SSDs and multiple sets of hard disk drives (HDDs), wherein each set of SSDs and HDDs store data arranged in multiple slices striped across the respective set of SSDs and HDDs;\npredicting, for each set of SSDs, a write rate at which data will be written to the multiple slices stored on the set of SSDs during a next time interval;\ndetermining, for each set of SSDs, a number of bytes that can be written to each set of SSDs, wherein the number is based on a remaining program /erase (PE) cycle count for each respective set of SSDs;\ndetermining, for each set of SSDs, a wear metric representative of a wear rate corresponding to the set of SSDs, the metric based on a SSDs' determined predicted write rate and the determined number of bytes that can be written to each set of SSDs;...",
    "description": "TECHNICAL FIELD\nThe present invention relates to managing solid state drive wear rate in hybrid data storage arrays.\nBACKGROUND OF THE INVENTION\nStorage devices are employed to store data that are accessed by computer systems. Examples of basic storage devices include volatile and non-volatile memory, floppy drives, hard disk drives, tape drives, and the like. A storage device may be locally attached to an input/output (IO) channel of a computer. For example, a hard disk drive may be connected to a computer's disk controller.Ms 420b-d each include two indicators of the primary criteria set, the DMs 420b-d are then further ranked based on the secondary criteria of I/O workload denoted by column 416 values. The higher the I/O workload, the higher the ranking of the proposed DM. Thus, in this example, assume X2>X3>X4 resulting in the ranking of 420b-d as in the example 400. Since DMs 420e-g do not meet any of the primary criteria, the DMs 420e-g are ranked lower than DMs 420a-d. DMs 420e-g are then ranked based on the secondary criteria of I/O workload denoted by column 416 values. The higher the I/O workload, the higher the ranking of the proposed DM. In this example, assume X5>X6>X7 thereby resulting in the ranking of 420e-g as in the example 400.\nThus, the proposed DMs 420a-g are first ranked based on the primary criteria, and then, for a set of DMs equally ranked based on primary criteria, the set is then ranked based on the secondary criteria...",
    "inventor": "Dalmatov, Nickolay A.",
    "applicant": "EMC IP Holding Company LLC",
    "ipc_class": "G06F3/06",
    "cpc_class": "G06F3/0616\nG06F3/0647\nG06F3/0653\nG06F3/0665\nG06F3/0685\nG06F2212/7208",
    "passage": "for use with other data storage arrays by other vendors and with other components than as described herein for purposes of example.\nThe data storage system 12 may be a data storage array including a plurality of data storage devices 16a-16n. The data storage devices 16a-16n may include one or more types of data storage devices such as, for example, one or more disk drives and/or one or more solid state drives (SSDs). An SSD is a data storage device that uses solid-state memory to store persistent data. An SSD using SRAM or DRAM, rather than flash memory, may also be referred to as a RAM drive. SSD may refer to solid state electronics devices as distinguished from electromechanical devices, such as hard drives, having moving parts. Flash memory-based SSDs (also referred to herein as “flash disk drives,” “flash storage drives”, or “flash drives”) are one type of SSD that contains no moving mechanical parts.\nThe flash devices may be constructed using nonvolatile semiconductor NAND"
    },
    {
        "position": "2",
        "rating": "Related",
        "image": "https://beta.ipscreener.com/img.php?id=US-8732396-B2-2.png",
        "note": null,
        "document_type": "patent",
        "patent_number": "US8732396",
        "kindcode": "B2",
        "publication_date": "2010-12-09",
```

kind-code: The kind codes are used to identify the type of patent publication. More information on this syntax is available at: www.wipo.org. Some of the most common kind codes are:

- A1 - Publ. of Application with search report
- A2 - Publ. of Application without search report
- B1 - Patent publication
- B2 - Patent after modification

publication-date: The publication date is the date on which a patent application/grant is first published. It is the date on which the document is made available to the public.

priority-date: Priority date refers to the earliest filing date in a family of patent applications.

title (array): This section includes the full title of the patent.

- **text:** It is the title text of the patent.

abstract (array): This is the summary describing the essence of the scope of a patent.

- **text:** It is the text content of the abstract of the patent.

claim (array): A claim defines exactly what is


```
                "priority_date": "2009-06-08",
                "title": "Method and apparatus for protecting the integrity of cached data in
a direct-attached storage (DAS) system",
                "abstract": "A DAS system that implements RAID technology is provided in
which an array of solid state disks (SSDs) that is external to the DAS controllers of the DAS
system is used by the DAS controllers as WB cache memory for performing WB caching
operations. Using the external SSD array as WB cache memory allows the DAS system to be fully
cache coherent without significantly increasing the complexity of the DAS system and without
increasing the amount of bandwidth that is utilized for performing caching operations. In
addition, using the external SSD array as WB cache memory obviates the need to mirror DAS
controllers.",
                "claim": "1. A direct-attached storage (DAS) system comprising:\nan array of
magnetic hard disk drives (HDDs);\nan array of solid state disks (SSDs); and\nat least first
and second DAS controllers connected to the array of HDDs and to the SSD array, each DAS
controller having a central processing unit (CPU), a local memory device, and an input/output
(I/O) interface device, wherein each of the DAS controllers is configured to perform a
caching algorithm that causes data received in the respective DAS controller to be
temporarily stored in a cache memory of the SSD array and subsequently stored in one or more
of the HDDs of the array of HDDs, wherein the data has metadata associated therewith, and
wherein the caching algorithms performed by the respective DAS controllers cause the data to
be stored in blocks in the SSD array, each block including a data integrity field (DIF),...",
                "description": "CROSS-REFERENCE TO RELATED APPLICATIONS\nThis application
claims priority to and the benefit of the filing date of a U.S. provisional patent
application that was filed on Jun. 8, 2009, having Ser. No. 61/268,055, entitled “METHOD TO
EFFICIENTLY USE SSD AS WB CACHE ELEMENT IN BOTH PRIVATE AND SHARED DAS CONFIGURATIONS”, which
is incorporated herein by reference in its entirety.\nTECHNICAL FIELD OF THE INVENTION\nThe
invention relates generally to data storage systems and, more particularly, to a method and
apparatus for protecting the integrity of cached data in a direct-attached storage (DAS)
system...",
                "inventor": "BERT LUCA",
                "applicant": "BERT LUCA\nLSI CORPORATION",
                "ipc_class": "G06F12/00",
                "cpc_class": "G06F12/0866\nG06F11/1008\nG06F12/0804\nG06F12/084\nG06F2212
/222\nG06F2212/262",
                "passage": "the data is striped across multiple SSDs of the SSD array 110.
If, for example, RAID level 1 is used, then when each of the DAS controllers 120 stores data
in cache memory in the SSD array 110, the data is replicated, or mirrored, in multiple SSDs
of the SSD array 110. If one of the SSDs of the SSD array 110 fails, the RAID level of
technology that is implemented with the SSD array 110 will allow the data to be recovered. In
this way, the DAS system 100 is fully cache coherent. The invention is not limited with
respect to the RAID level that is used to ensure cache coherency for data that is cached in
the cache memory of the SSD array 110.\nAlso, the SSD array 110 is typically, but not
necessarily, partitioned into respective portions that are used by the respective DAS
controllers 120. For example, assuming there are a total of N DAS controllers 120, where N is
a positive integer that is equal to or greater than 1, the storage capacity of the SSD array
110 will be divided into N equal portions,"
            }
```


When no ranking or only noise has been assigned to documents. It still consider it as a successful response and return and empty request.

No rankings made


```
{
    "index": [
        {
            "id": "1",
            "name": "Master"
        },
        {
            "id": "999",
            "name": "Novelty screening"
        },
        {
            "id": "54523",
            "name": "Infringement screening"
        }
    ],
    "result": {
```

claimed by the invention and therefore what is sought to be protected. It clearly lays down what the patent does and does not cover.

- **text:** It is the text content of the claim of the patent.

description (array): The detailed description describes in detail what the invention is and how it is made and used. It reflects the complete picture of the invention.

- **text:** It is the text content of the description of the patent.

inventor (array): T his field provides information about the inventor(s).

- **name:** This field returns the name of the inventor (s).

applicant (array): This field provides information about the patent owner (s) or applicant(s).

- **name:** This field returns the name of the patent owner(s) or applicant(s).

class (array): The classification scheme is a system of codes that groups inventions according to technical area, where IPC and CPC is the most common. The class information is divided into the follwoing hierarchy, including four sections:

- **sub:** This is the complete class

```
        "1": [],
        "999": [],
        "54523": []
    }
}
```

Error Response

The below response will be returned if your **API key** is wrong or invalid.

API key invalid/wrong

```
{
    "status": "error",
    "message": "API key is not valid"
}
```

The below response will be returned if your **case id** is wrong, invalid or when you try to open a case id that belongs to another customer/company

No permission

```
{
    "status": "error",
    "message": "You dont have permission to do that"
}
```

Sample Call (cURL)

Sample Call (cURL)

```
curl --location --request GET 'https://api.ipscreener.com/v1/idea' \
--header 'key: [value]' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'case= 20718'
```

information e. g. H04M15 /03.
- **type:** This declares the classification system referred to e. g. IPC, CPC

passage: Shows the paragraph within a document that the AutoMatch engine considered to be most relevant to the query.

- **section**: The section where the relevant para graph is located
- **text**: The paragraph within a document considered to be most relevant by the engine.