API Documentation for NVD-CVE

The API enables functionalities to fetch CVE data, filter results based on different parameters, and synchronize the database periodically.

The following documentation outlines the available API endpoints, their methods,parameters and responses. This ensures seamless integration, enabling easy access to up-to-date vulnerability information.

API Endpoints

1.Fetch All CVEs

URL:
http://localhost:3000/cves/list

Method:
GET

Query Parameters:

- limit : Number of records to fetch per page. Default is 10.
- page : Page number for paginated results. Default is 1.
- sortOrder : Sorting order of the results (asc or desc). Default is desc.
- sortField : Field to sort the results by. Default is publishedDate.

Request:

curl --location
'http://localhost:3000/cves/list?limit=10&page=1&sortOrder=desc&sortField=publishedDate'

Response:


```
{
  "success": true,
  "data": [
    {
      "metrics": {
        "cvssMetricV2": {
          "cvssData": {
            "baseScore": 4,
            "vectorString": "AV:N/AC:L/Au:S/C:P/I:N/A:N",
            "accessVector": "NETWORK",
            "accessComplexity": "LOW",
```

```
        "authentication": "SINGLE",
        "confidentialityImpact": "PARTIAL",
        "integrityImpact": "NONE",
        "availabilityImpact": "NONE"
       }
      },
      "cvssMetricV31": {
       "cvssData": {
        "baseScore": 4.3,
        "vectorString": "CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N",
        "attackVector": "NETWORK",
        "attackComplexity": "LOW",
        "privilegesRequired": "LOW",
        "userInteraction": "NONE",
        "scope": "UNCHANGED",
        "confidentialityImpact": "LOW",
        "integrityImpact": "NONE",
        "availabilityImpact": "NONE"
       }
      }
     },
     "cveId": "CVE-2024-13042",
     "sourceIdentifier": "cna@vuldb.com",
     "publishedDate": "2024-12-30T15:45:06.523Z",
     "lastModifiedDate": "2024-12-30T15:45:06.523Z",
     "descriptions": [
      {
       "lang": "en",
       "value": "A vulnerability was found in Tsinghua Unigroup Electronic Archives Management System..."
      }
```

```
    ]
   }
 ],
 "totalRecords": 200,
 "currentPage": 1,
 "totalPages": 2
}
```


2.Fetch CVE by ID

URL:
http://localhost:3000/cves/:cveId

Method:
GET

Path Parameters:

- cveId : The unique identifier of the CVE to fetch.

Request:

curl --location 'http://localhost:3000/cves/CVE-2024-13042'

Response:


```
{
 "metrics": {
  "cvssMetricV2": {
   "cvssData": {
    "baseScore": 4,
    "vectorString": "AV:N/AC:L/Au:S/C:P/I:N/A:N",
    "accessVector": "NETWORK",
    "accessComplexity": "LOW",
    "authentication": "SINGLE",
    "confidentialityImpact": "PARTIAL",
    "integrityImpact": "NONE",
    "availabilityImpact": "NONE"
   }
  },
```

```
    "cvssMetricV31": {
      "cvssData": {
        "baseScore": 4.3,
        "vectorString": "CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N",
        "attackVector": "NETWORK",
        "attackComplexity": "LOW",
        "privilegesRequired": "LOW",
        "userInteraction": "NONE",
        "scope": "UNCHANGED",
        "confidentialityImpact": "LOW",
        "integrityImpact": "NONE",
        "availabilityImpact": "NONE"
      }
    }
  },
  "cveId": "CVE-2024-13042",
  "sourceIdentifier": "cna@vuldb.com",
  "publishedDate": "2024-12-30T15:45:06.523Z",
  "lastModifiedDate": "2024-12-30T15:45:06.523Z",
  "descriptions": [
    {
      "lang": "en",
      "value": "A vulnerability was found in Tsinghua Unigroup Electronic Archives Management System..."
    }
  ],
  "weaknesses": [
    {
      "source": "cna@vuldb.com",
      "type": "Primary",
      "description": [
```

```
      {
        "lang": "en",
        "value": "CWE-200"
      },
      {
        "lang": "en",
        "value": "CWE-284"
      }
    ]
  }
]
```


Unit Tests

Test 1: Test GET /cves/list

Test Case:

```
def test_get_cves_list():
    response = client.get("/cves/list?startIndex=0&resultsPerPage=10")
    assert response.status_code == 200
    assert "totalRecords" in response.json
    assert "cves" in response.json
    assert len(response.json["cves"]) == 10
```

Output:


```
{
    "totalRecords": 277633,
    "cves": [
        {"cveID": "CVE-2023-0001", "description": "description 1", "cvssScore": 5.4},
        {"cveID": "CVE-2023-0002", "description": "description 2", "cvssScore": 7.2},
    ]
}
```

Invalid Input:

```
def test_get_cves_list_invalid_page_size():
    response = client.get("/cves/list?startIndex=0&resultsPerPage=10")
    assert response.status_code == 40;
```

Output:

```
{
    "error": "resultsPerPage cannot exceed 10"
}
```

Test 2: Test GET /cves/{cveID}

Valid CVE ID:

```
def test_get_cve_by_id():
    cve_id = "CVE-2023-12345"
    response = client.get(f"/cves/{cve_id}")
    assert response.status_code == 200
```

Output:

```
{
    "cveID": "CVE-2023-12345",
    "description": "A sample vulnerability description.",
    "cvssScore": 9.8,
    "publishedDate": "2023-06-01"
}
```

Invalid CVE ID:

```
def test_get_cve_by_invalid_id():
    response = client.get("/cves/CVE-9999-9999")
    assert response.status_code == 404
```

Output:

```
{
    "error": "CVE ID not found"
}
```

Test 3: Test POST /sync/refresh

```
def test_data_sync_full_refresh():
    response = client.post("/sync/refresh?type=full")
    assert response.status_code == 200
```

Output:

```
{
    "message": "Data refresh complete",
    "recordsSynced": 10
}
```

Test 4: Test Data Cleansing

```
def test_data_cleansing():
    response = client.get("/cves/list?startIndex=0&resultsPerPage=10")
    cves = response.json["cves"]
    assert len(cves) == len(set([cve["cveID"] for cve in cves]))
```

Output:

```
{
    "totalRecords": 277633,
    "cves": [
        {"cveID": "CVE-2023-0001", "description": "description 1", "cvssScore": 5.4},
        {"cveID": "CVE-2023-0002", "description": "description 2", "cvssScore": 7.2},
        {"cveID": "CVE-2023-0001", "description": "Duplicate CVE", "cvssScore": 4.1}
    ]
}
```