

## Problem Statement:

[https://github.com/codingo/Ransomware-Json-Dataset/blob/master/ransomware\\_overview.json](https://github.com/codingo/Ransomware-Json-Dataset/blob/master/ransomware_overview.json)

Given this JSON as a sample, how will you write a short script/program to process this json and store into databases

Explain selection of coding language and DB of your choice.

In the case of large set of data, how will you ensure that no records are missed when writing into the DB

In the case of duplicate entries, how will you handle them?

Design and Implement a REST API for front end consumption

Design a RESTful API for the system that allows users to store and retrieve data. Implement the endpoints in the language of your choice. Ensure that your API supports the basic CRUD operations on the data.

Design UI for the system for reporting dashboard

## Solution:

**Part1: Explain selection of coding language and DB of your choice.**

**Ans:** Ideally, we would use Python + MongoDB as Python has good JSON processing libraries and integration with MongoDB. We use a non-relational database like MongoDB as the dataset is regular and has sub-types like Arrays embedded within the primary JSON object. Non-relational database offers better performance for large data sets.

Since, I do not have working experience in Python or MongoDB, I am going with the next best alternative JAVA springboot application with Elasticsearch as the preferred DB of choice due to my familiarity with it. Elastic search is also a NoSQL database that stores data as documents and is optimized for searching.

**Part2: In the case of large set of data, how will you ensure that no records are missed when writing into the DB**

**Ans:** For importing large datasets, we have the bulk indexing option provided by Elastic Search this will ensure that documents are written in an optimized manner without missing any and will gracefully handle errors.

### Part3: In the case of duplicate entries, how will you handle them?

**Ans:** Duplicate entries can be avoided in Elastic search by using a unique identifier as the id of the document. In this case, I am going to use the name property in the JSON object. If there are multiple names, the id will be a combination of the names in the array separated by a "~" symbol. In the below example, the id will be 777~Sevleg.

```
{
  "name": [
    "777",
    "Sevleg"
  ],
  "extensions": ".777",
  "extensionPattern": "._[timestamp]_${email}$.777\\ne.g. ._14-05-2016-11-59-36_${ninja.gaiver@aol.com}$.777",
  "ransomNoteFileNames": "read_this_file.txt",
  "comment": "",
  "encryptionAlgorithm": "XOR",
  "decryptor": "",
  "resources": [
    "https://decrypter.emsisoft.com/777"
  ],
  "screenshots": "",
  "microsoftDetectionName": "Ransom:Win32/Empercrypt.A",
  "microsoftInfo":
"https://www.microsoft.com/security/portal/threat/Encyclopedia/Entry.aspx?Name=Ransom:Win32/Empercrypt.A",
  "sandbox": "https://www.hybrid-analysis.com/sample/2955d081ed9bca764f5037728125a7487f29925956f3095c58035919d50290b5?environmentId=4",
```

```

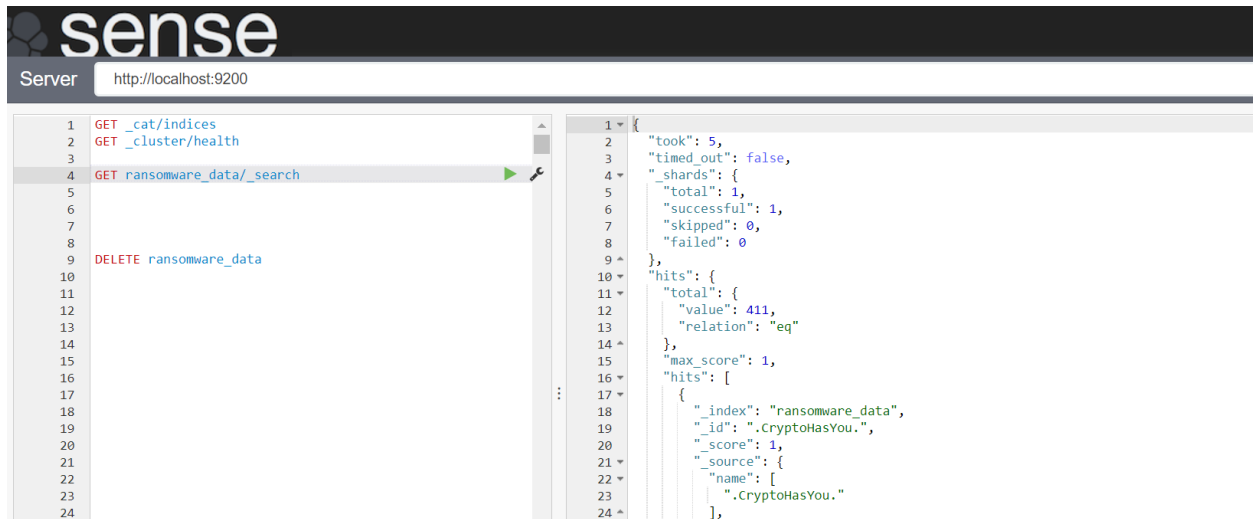
    "iocs": "https://otx.alienvault.com/pulse/573b02701116a040ceccdd85/",

    "snort": ""

}

```

The Code solution for Part1-3 is found in the repository under ransomware-data folder. Please find the screenshot below for the data getting successfully inserted in the database:



```

    "snort": ""
  },
  {
    "index": "ransomware_data",
    "id": "777~Sevleg",
    "score": 1,
    "source": {
      "name": [
        "777",
        "Sevleg"
      ],
      "extensions": ".777",
      "extensionPattern": "._[timestamp]_[email]$.777\\ne.g. ._.14-05-2016-11-59-36_$ninja.gaiver@aol.com$.777",
      "ransomNoteFileNames": "read_this_file.txt",
      "comment": "",
      "encryptionAlgorithm": "XOR",
      "decryptor": "",
      "resources": [
        "https://decrypter.emsisoft.com/777"
      ],
      "screenshots": "",
      "microsoftDetectionName": "Ransom:Win32/Empercrypt.A",
      "microsoftInfo": "https://www.microsoft.com/security/portal/threat/Encyclopedia/Entry.aspx?Name=Ransom:Win32/Empercrypt.A",
      "sandbox": "https://www.hybrid-analysis.com/sample/2955d081ed9bca764f5037728125a7487f29925956f3095c58035919c",
      "iocs": "https://otx.alienvault.com/pulse/573b02701116a040ceccdd85/",
      "snort": ""
    }
  },
  {
    "index": "ransomware_data",
    "id": "7ev3n~/ev3n-HONE$T",
    "score": 1,
    "source": {
      "name": [
        "7ev3n",
        "7ev3n HONE$T"
      ]
    }
  }
]

```

#### Part4: Design and Implement a REST API for front end consumption

Design a RESTful API for the system that allows users to store and retrieve data. Implement the endpoints in the language of your choice. Ensure that your API supports the basic CRUD operations on the data.

**Ans:** I am using a spring boot application to design the REST API and it supports all the CRUD operations. The code for this can be found under **ransomware** folder in the repository.

Create:

HTTP <http://localhost:8081/api/ransomware> Save Share

**POST** <http://localhost:8081/api/ransomware> Send

Params Authorization Headers (8) **Body** Scripts Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

```
1 {
2   "name": "Sankar",
3   "extensions": ".enc",
4   "extensionPattern": "",
5   "ransomNoteFileNames": "YOUR FILES ARE LOCKED.txt".
6 }
7
```

Body Cookies Headers (5) Test Results 200 OK • 75 ms • 169 B • 🌐 🔍 ⋮

Pretty Raw Preview Visualize Text 🔍

1 Sankar

← → ↺ [localhost:5601/app/sense](http://localhost:5601/app/sense) ★ 🔍 ⋮

# sense

Server <http://localhost:9200> 🔍 ⚙️

```
1 GET _cat/indices
2 GET _cluster/health
3
4 GET _search
5 {
6   "query": {
7     "match": {
8       "_id": "Sankar"
9     }
10  }
11 }
12
13
14
15
16 DELETE _search
17
18
19
20
21
22
23
24
25
26
27
```

```
8   "failed": 0
9 },
10 "hits": {
11   "total": {
12     "value": 1,
13     "relation": "eq"
14   },
15   "max_score": 1,
16   "hits": [
17     {
18       "_index": "ransomware_data",
19       "_id": "Sankar",
20       "_score": 1,
21       "_source": {
22         "name": {
23           "Sankar"
24         },
25         "extensions": ".enc",
26         "extensionPattern": "",
27         "ransomNoteFileNames": "YOUR FILES ARE LOCKED.txt",
28         "comment": "",
29         "encryptionAlgorithm": "AES(256)",
30         "decryptor": "",
31         "resources": [
32           "http://www.nyxbone.com/malware/CryptoHasYou.html"
33         ],
34         "ransomNote": ""
35       }
36     }
37   ]
38 }
```

Read:

localhost:8081/api/ransomware/.CryptoHasYou.

Pretty-print ☒

```
{
  "name": [
    ".CryptoHasYou."
  ],
  "extensions": ".enc",
  "extensionPattern": "",
  "ransomNoteFileNames": " YOUR_FILES_ARE_LOCKED.txt",
  "comment": "",
  "encryptionAlgorithm": "AES(256)",
  "decryptor": "",
  "resources": [
    "http://www.nyxbone.com/malware/CryptoHasYou.html"
  ],
  "screenshots": "",
  "microsoftDetectionName": "Trojan:Win32/Dynamer!ac",
  "microsoftInfo": "https://www.microsoft.com/security/portal/threat/encyclopedia/entry.aspx?Name=Trojan%3AWin32%2FDynamer!ac",
  "sandbox": "https://www.hybrid-analysis.com/sample/afd3394fb538b36d20085504b86000ea3969e0ae5da8e0c058801020ec8da67c?environmentId=4",
  "iocs": "https://otx.alienvault.com/pulse/57180b18c1492d015c14bed8/",
  "snort": ""
}
```

## Update:

PUT http://localhost:8081/api/ransomware/Sankar Send

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
4 ],
5 "extensions": ".enc",
6 "extensionPattern": "",
7 "ransomNoteFileNames": "YOUR_FILES_ARE_UNLOCKED.txt",
8 "comment": "",
9 "encryptionAlgorithm": "AES(256)",
10 "decryptor": "",
```

Body Cookies Headers (4) Test Results 200 OK • 54 ms • 123 B

localhost:5601/app/sense

sense

Server http://localhost:9200

```
1 GET _cat/indices
2 GET _cluster/health
3
4 GET ransomware_data/_search
5 {
6   "query": {
7     "match": {
8       "_id": "Sankar"
9     }
10  }
11 }
12
13
14
15
16 DELETE ransomware_data
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
```

```
{
  "max_score": 1,
  "hits": [
    {
      "_index": "ransomware_data",
      "_id": "Sankar",
      "_score": 1,
      "_source": {
        "name": [
          "Sankar"
        ],
        "extensions": ".enc",
        "extensionPattern": "",
        "ransomNoteFileNames": "YOUR_FILES_ARE_UNLOCKED.txt",
        "comment": "",
        "encryptionAlgorithm": "AES(256)",
        "decryptor": "",
        "resources": [
          "http://www.nyxbone.com/malware/CryptoHasYou.html"
        ],
        "screenshots": ""
      }
    }
  ]
}
```

## Delete:

DELETE

http://localhost:8081/api/ransomware/Sankar

Send

Params

Authorization

Headers (8)

Body

Scripts

Tests

Settings

Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (4)

Test Results

200 OK • 48 ms • 123 B •

← → ↺

localhost:5601/app/sense

Server http://localhost:9200

```

1 GET _cat/indices
2 GET _cluster/health
3
4 GET ransomware_data/_search
5 {
6   "query": {
7     "match": {
8       "_id": "Sankar"
9     }
10  }
11 }
12
13
14
15
16 DELETE ransomware_data
17
18
19
20
21

```

```

1 {
2   "took": 7,
3   "timed_out": false,
4   "_shards": {
5     "total": 1,
6     "successful": 1,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": {
12      "value": 0,
13      "relation": "eq"
14    },
15    "max_score": null,
16    "hits": []
17  }
18 }

```

Also, implemented the API with pagination:

← → ↺

localhost:8081/api/ransomware?page=2&size=10

☆ {}

pretty-print

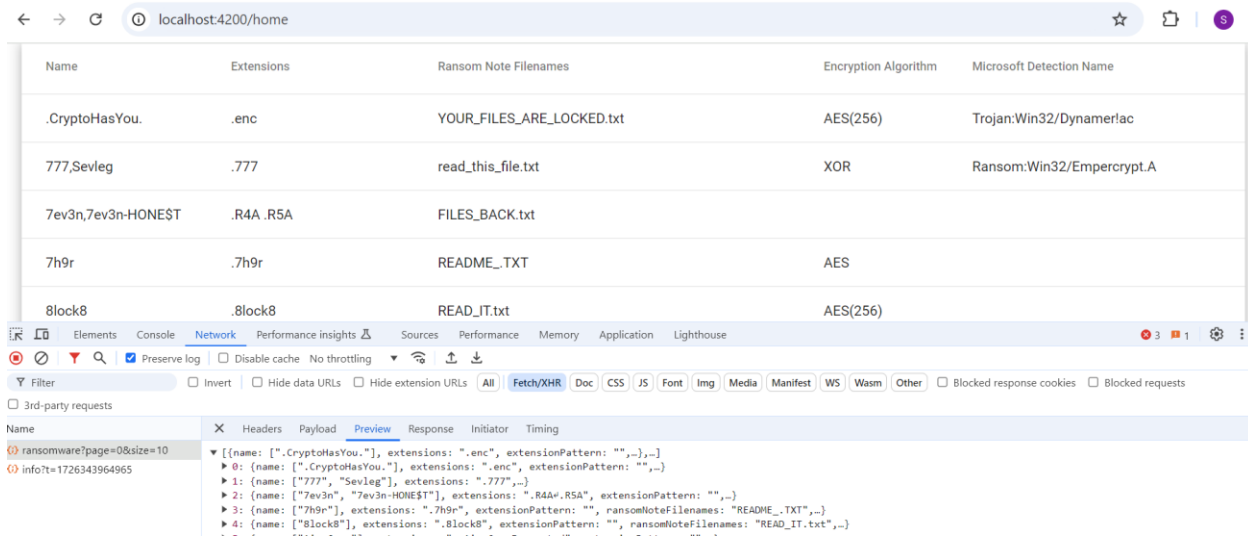
```

{
  "name": [
    "Bandarchor",
    "Rakhni"
  ],
  "extensions": ".id-1235240425_help@decryptservice.info",
  "extensionPattern": ".id-[ID]_[EMAIL_ADDRESS]",
  "ransomNoteFileNames": "HOW TO DECRYPT.txt",
  "comment": "Files might be partially encrypted",
  "encryptionAlgorithm": "AES(256)",
  "decryptor": "",
  "resources": [
    "https://reaqta.com/2016/03/bandarchor-ransomware-still-active/"
  ],
  "screenshots": "https://www.bleepingcomputer.com/news/security/new-bandarchor-ransomware-variant-spreads-via-malvertising-on-adult-sites/"
},
{
  "name": [
    "BarRax"
  ],
  "extensions": ".BarRax",
  "extensionPattern": "",
  "ransomNoteFileNames": "",
  "comment": "Based on HiddenTear",
  "encryptionAlgorithm": "",
  "decryptor": "",
  "resources": [
    "https://twitter.com/demonslay335/status/835668540367777792"
  ],
  "screenshots": ""
}

```

## Part5: Design UI for the system for reporting dashboard

**Ans:** Basic UI is implemented with Angular 12 to display a few selected columns from the database. The code for this can be found in **ransomware-ui** folder in the repository. Screenshot below:



## Notes/Considerations:

1. No security or authentications mechanisms like JWT has been implemented.
2. No data validations done from UI or Backend side.
3. Exhaustive exception handling or error code handling has not been implemented.