

Recover Data about Detected Defects of Underground Metal Elements of Constructions in Amazon Elasticsearch Service

Roman Mysiuk¹, Volodymyr Yuzevych²

¹ *Ivan Franko National University of Lviv*

1 Universytetska Street, Lviv, 79000, Ukraine

² *Karpenko Physico-mechanical Institute of the NAS of Ukraine*

5 Naukova Street, Lviv, 79060, Ukraine

DOI: [10.22178/pos.89-9](https://doi.org/10.22178/pos.89-9)

JEL Classification: L95

Received 25.12.2022

Accepted 27.01.2023

Published online 31.01.2023

Corresponding Author:

Roman Mysiuk

mysyukr1@ukr.net

© 2023 The Authors. This article is licensed under a [Creative Commons Attribution 4.0 License](https://creativecommons.org/licenses/by/4.0/)



Abstract. This paper examines data manipulation in terms of data recovery using cloud computing and search engine. Accidental deletion or problems with the remote service cause information loss. This case has unpredictable consequences, as data must be re-collected. In some cases, this is not possible due to system features. The primary purpose of this work is to offer solutions for received data on detected defects of underground metal structural elements using modern information technologies.

The main factors that affect underground metal structural elements' durability are the soil environment's external action and constant maintenance-free use. Defects can usually occur in several places, so control must be carried out along the entire length of the underground network. To avoid the loss of essential data, approaches for recovery using Amazon Web Service and a developed web service based on the REST architecture are considered. The general algorithm of the system is proposed in work to collect and monitor data on defects of underground metal structural elements. The result of the study for the possibility of data recovery using automatic snapshots or backup data duplication for the developed system.

Keywords: recovering data; search; cloud computing; snapshots; corrosion; defects; underground metal elements of construction.

INTRODUCTION

Underground metal elements of constructions provide transportation and delivery of oil and gas products to different countries of Eastern and Central Europe, which is 2–3 times faster than in any other way. With constant operation in underground metal elements of constructions, there are various defects due to active mechanical stress and function in specific conditions of the soil environment. Severe defects usually cause accidents on such lines and lead to environmental and even man-made disasters. Based on the possible risks, all underground metal elements of constructions are carefully monitored and maintained in good working order so that no cracks can affect the functioning of the structure [1, 2]. To monitor such systems, a large amount of data needs to be processed because the length of underground metal elements of

constructions is kilometres [3]. Big Data is actively developing in Ukraine and is used in various sectors of the economy due to the rapid scalability of such systems. The most popular way to transfer data is to use network data transfer protocols. Elasticsearch is a non-relational database (NoSQL) in which data is transmitted via network requests. It can be used as a portable microcomputer with the possibility of using additional modules to collect input information for the system. One of the most popular cloud environments is Amazon Web Service (AWS). This allows constant access to the database and data processing using cloud technology. The work aims to create a secure system that can analyse input information and show the possibility of data recovery in case of loss in such scenarios.

MATERIALS AND METHODS

Materials and Methods

There are various types of databases used for storing different varieties of data:

- A centralised database differs from others in that it is accessed through a computer network, which can give them access to a central processing unit (CPU) that supports the database. Such a means of using databases are commonly used in local networks.
- A distributed database is a database that consists of two or more files located on different sites or the same network, or completely different networks. Parts of the database are stored in several physical locations, and processing is distributed among many database nodes.
- NoSQL database, which provides a mechanism for storing and retrieving data, is different from the approach of relationship tables in relational databases. NoSQL databases are increasingly used in big data tasks and real-time web applications.
- A relational database is a collection of data elements organised as a set of formally described tables, from which data can be accessed or re-collected in many ways without reorganising the database tables.
- A cloud database is a type of database that performs queries on a remote machine.
- In computing, a graph database uses various nodes, edges, and properties to represent and store information for semantic queries. This type is used to visualise complex structural connections better.

Cloud technologies and computing are becoming increasingly popular. This is because cloud environments have many useful features and security. Amazon Web Service is chosen for the developed system due to its user-friendly interface and easy integration.

This system uses a non-relational database because NoSQL databases are document, key-value, graph, or wide-column stores and are well suited for working with many JavaScript Object Notation (JSON) documents.

Elasticsearch (ES) is a search engine that uses the Lucene library. This allows you to create a distributed full-text search with the ability to use multiple units using the Hypertext Transfer Protocol (HTTP) web interface and schema-free

JSON documents [4]. Elasticsearch provides a full Query Domain-Specific Language DSL (DSL) based on JSON to define queries. Elastic search queries can include many conditions simultaneously, and the size of such a query can be significant. Queries can be divided into geo, compound, range, term level, string, multi-match, full text, and match all [5].

In ES, all data is contained in separate indexes, and searches can join using a query with multiple fields at once [6]. Indexes can be divided into shards, and each shard can have any number of replicas. Routing and rebalancing operations are performed automatically when new documents are added. This database has a separate bulk Application Programming Interface (API) function for data ingestion, which allows multiple operations to index and deletes in a single API request. It can significantly increase the indexing speed. To execute bulk API, we need to have a pre-formed data file and run it from the command line using a simple curl command. There are some restrictions on file size and the number of entries. It has been tested that the file should not exceed about 50 MB or the number of records of 100,000 for the presented case [7].

Amazon Elasticsearch Service (Amazon ES) lets you store up to 3 PB of data in a single cluster, enabling you to run large log analytics workloads via a single Kibana interface. Using the AWS cloud environment allows us to have a global, highly accessible, secure domain to access ES [8]. Another advantage is the quick-to-search data [9].

Data recovery is one of the main issues [10]. There is a unique mechanism for recovering data lost or deleted, called snapshots. Snapshots are backups of the Amazon ES cluster indexes and status. The state includes cluster parameters, node information, index parameters, and shard distribution. Data unavailability may be due to index status. Possible positions include green, yellow, and red. The last two signal some problems; in this case, data recovery is also essential. Amazon Elasticsearch is ready to recover data [11]:

- Automated snapshots are for cluster recovery only. Amazon ES stores automated snapshots in a pre-configured bucket of AWS Simple Cloud Storage (S3) at no extra charge.
- Manual snapshots are used to restore a cluster or to move data from one cluster to another. This

requires manual shooting. These snapshots usually use your Amazon S3 bucket, and S3 payments apply accordingly.

Experimental procedures

This section describes the system's main configuration, data preparation, and design. The processes of setting up the cloud environment, filling it with data, and the structure of the web service are described.

System Configuration. A restful web service is created that works with ES and displays information about the status of underground metal elements of constructions in real time. This web service is developed using Java programming language and Spring framework [12].

Before starting with our approach, you should follow a few preliminary steps. The following tools are used in the development:

- Java Development Kit – a set of development tools and libraries containing documentation, standard libraries, and code compiler for Java classes. Today, the approach of building a client-server architecture using the Java programming language through cross-platform and object-oriented language is prevalent.
- Maven is an open-source programming tool for building our projects. There are a few phases of the lifecycle: validate, compile, test, package, verify, install, and deploy. Each of these phases helps to perform specific actions with the code. This application construction allows using ready-made functions to collect code into one assembly. Moreover, all additional libraries are downloaded from a remote repository, which replaces jar files.

To efficiently run multiple instances of Elasticsearch locally, you need to download the tar or zip archive from the official site and run the bat file in the bin folder.

To test ES in the cloud, follow these steps using the Amazon ES, AWS Command Line Interface (AWS CLI), or AWS Software Development Kit (AWS SDK):

1. Create an Amazon ES domain. To do this, you must select the appropriate general settings, such as deployment and instance types, and provide a name for the domain, etc. All other settings can be ignored for now. In my case, one of the smallest instance types was selected

t3.small.elasticsearch. According to on-demand instance pricing for this type, vCPU, and Memory (GiB) are 2. Which is enough for initial testing [8].

2. Wait until the settings are activated. It typically takes 15–20 minutes to initialise.

3. Upload the data to the Amazon ES domain for indexing.

Data Preparation. For the current testing of this system, the generated test data using Bash were used randomly. The Restful service mocking feature allows us to simulate accurate data using generated test data. In the future, there is a plan to integrate actual test data from some external devices.

Bash is a scripting language where any command can be grouped and executed in a single file. Functions can be created in any programming language, simplifying code reuse. The bash script is a temporary solution. Instead of fundamental values, random values of characters are generated using functions in Bash. The main steps are following in this script:

- Create variables for index names for devices and ES domains.
- Create a mapping of records in ES for each index using a loop [13]. The default mapping is created at the first record, but when you want to make a custom. This is filled in once and is editable. Mapping indicates in what format and type of data will be stored. The curl tool sends requests via the command line, which allows you to work with URL syntax. The following command can be executed:

```
curl -X PUT "$domain/$indexName" -H "Content-Type: application/json" -H "Accept: application/json" -d @$deviceIndexMapping
```

- Generate a temporary file with test data filled with random values using Bash. Figure 1 displays a model of test data. The JSON format, which is presented in a key-value format, was used to describe the data. This presentation is easy to read and acceptable for Elasticsearch search engines.
- Push data from file to ES using Bulk API:

```
curl -H 'Content-Type: application/x-ndjson' -XPOST $domain/_bulk --data-binary @"generated_data.json"
```

– Remove the temporary file.

Generated files should be divided and pushed in a loop due to the limitation of bulk API operations. For each index to have approximately the same number of records, the number of completed cycles must be multiplied by the number of records per push and divided by the number of indices. As a result, it allows the approximate number of iterations of loops to reach a certain number in each index. There is also a collection of information on various parameters, such as time and relativity to the device, allowing you to display it on web applications.

```
{
  "id": "'$randomId'",
  "device_name": "'${($RANDOM % 1000 + 1)}'",
  "date": "'date +%D'",
  "time": "'date +%T'",
  "external_factors": [
    {
      "soil_moisture": "'${($RANDOM % 100 + 1)}'",
      "temperature": "'${($RANDOM % 100 + 1)}'"
    }
  ],
  "potential_current": [
    {
      "value": "'${($RANDOM % 1000 + 1)}'"
    }
  ],
  "geographic_coordinate_system": [
    {
      "latitude": "'${($RANDOM % 1000 + 1)}'",
      "longitude": "'${($RANDOM % 1000 + 1)}'"
    }
  ]
}
```

Figure 1 – Example of test data filled with random values using Bash script

System design. The Internet is a universal way to access data anywhere in the world. The development of web services as a backend part of the browser has become very popular due to the increasing use of web browsers [14].

The structure of the proposed system is shown in Figure 2. The central part of the system is a web service. The most used architectural style REST is used for creation. This web service can easily obtain and display the desired data. The service is developed based on Spring Framework.

To avoid external data interception and make the system more closed, it is recommended to use the secure Hypertext Transfer Protocol Secure. With Transport Layer Security (TLS) or Secure Sockets Layer (SSL), data is encrypted with this

approach. Therefore, you need to configure environment variables in the settings or the environment of the program or computer with which the web service works.

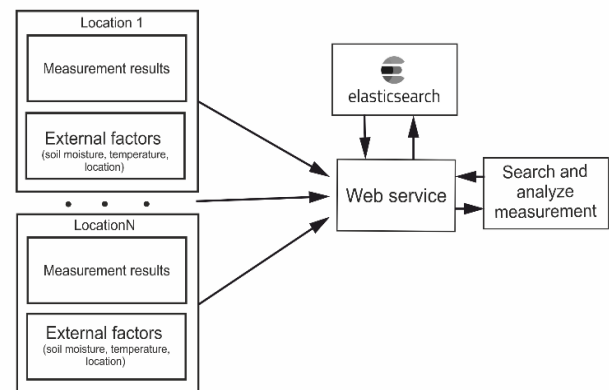


Figure 2 – The system for collecting and analysis for underground metal elements of constructions architecture

Although personnel data is not used in this system, we must take care of our product's security by reducing the system's vulnerability to external attacks. The approach of using Open Authorization (OAuth) for greater security is also popular, which is added to the existing web service too. With the help of generated tokens with a request, communication with the system can be sent.

Several endpoints are developed in web service for restoring data using some from the list of standard HTTPS methods such as GET, HEAD, POST, PUT, PATCH, DELETE, CONNECT, OPTIONS, and TRACE. For instance: GET method with request template:

`{domain}/snapshots/{snapshot_id}`, POST `{domain}/snapshots` with the following request body `“{“from”:“from_snapshot_id”, “to”:“to_snapshot_id”}”` and DELETE `{domain}/snapshots/{snapshot_id}`.

This group of operations is based on the Reindex API in ES. This feature is an additional option for data recovery, although it can cause redundancy of duplicate data.

In addition, the system allows getting information on time, id, device, and all the information, in general, using the web service. The following endpoints have been created for getting information:

- GET {domain}/measurement/devices.
- GET {domain}/measurement/devices/{device_id}.
- GET {domain}/measurement/{measurement_id}.
- GET {domain}/measurement/datatime?timeFrom={startTime}&timeTo={finishTime}.

This system can be considered a cyber-physical system. A new method for assessing the degree of the information security risk of the system for the control of underground metal constructions is proposed. In addition, taking into account the index of the probability of a successful attack on the system, the index of the impact of the attack, the adjustment index, which provides feedback, and the quality criterion [15].

Web service works like an additional layer on the system. In this case, that helps us to communi-

cate with data using a particular query. From on security perspective, that allows preventing different external attached to the system in further testing.

RESULTS AND DISCUSSION

Before starting the web service, the environment variables must be configured for setting OAuth token and using HTTPS, then:

- 1) Run Bash script to generate and push test data in Elasticsearch.
- 2) Check created indices in Figure 3. The created indexes are presented in the form of a table with the size of the existing data, the current state of the cluster, and other information. To view this information, the `/_cat/indices` endpoint of Elasticsearch must be reached.
- 3) Run the project using the command line: `mvn clean install`.

health	status	index	uuid	pri	rep	docs.count	docs.deleted	store.size	pri.store.size
green	open	device-19	SycDby8V50CQFKYA1M1B0Q	1	0	3699	0	789.3kb	789.3kb
green	open	device-18	cmw8ob47Tzuzf3dyng5Qiw	1	0	994	0	386.3kb	386.3kb
green	open	device-17	1IZFvK-vQ1SZtG-nQQyuUg	1	0	1061	0	390.8kb	390.8kb
green	open	device-16	vOKTBohSTFusukyXHLrSPg	1	0	1056	0	398.8kb	398.8kb
green	open	device-15	Wb8em6ikQG6GSAIomChWmg	1	0	979	0	383kb	383kb
green	open	device-14	Mt_SHR6gRwyP_Gxu8ZH84g	1	0	1065	0	400.3kb	400.3kb
green	open	device-13	RdvvTeDURtCtMukuWnYQRQ	1	0	1016	0	399.5kb	399.5kb
green	open	device-12	ifuMB1CKRt2du5_DOMQC8Q	1	0	1051	0	397.8kb	397.8kb
green	open	device-11	Ux-fl4j1Q3mvQms4zGWMFA	1	0	1035	0	402.3kb	402.3kb
green	open	device-10	5M1ZvR23TN6TK1EZ8r4j4g	1	0	1007	0	388.5kb	388.5kb
green	open	device-50	_apu0d4lQPkvjuvuvHp6KQ	1	0	3718	0	734.6kb	734.6kb
green	open	device-7	OIHfV_TSSRGkL3G1KEHqJA	1	0	1020	0	391.5kb	391.5kb
green	open	device-8	C116YtcVTaunH8hXMaEfQ	1	0	995	0	397.7kb	397.7kb
green	open	device-9	KWjTL1N1Rry5SDJgEvJEpw	1	0	1008	0	398.1kb	398.1kb
green	open	device-49	867h3hJrSnCTgyT9AHH6Jg	1	0	1034	0	402.7kb	402.7kb
green	open	device-48	QiZ2aULvRR6LbYd18Mxtww	1	0	1035	0	394.4kb	394.4kb
green	open	device-47	cLUt21o_S3uSSqV4-nfnzw	1	0	1041	0	395.4kb	395.4kb
green	open	device-46	VLb1WNR4T-OIsdNaAFmETg	1	0	1051	0	397.5kb	397.5kb
green	open	device-45	XkVeaRtCStGwGb1R6eYttw	1	0	1086	0	416.5kb	416.5kb
green	open	device-44	i-s8liuiSCGPrFma98HTw	1	0	1073	0	413kb	413kb
green	open	device-43	D1tumivkRO62W0MSRI6kCA	1	0	1014	0	398.4kb	398.4kb
green	open	device-42	ypqr5jxtQx6oeo-1W4SR6A	1	0	1032	0	402.5kb	402.5kb
green	open	device-41	7tC1ZuTEQAKK-1SpaZ1ZiA	1	0	3795	0	770.4kb	770.4kb
green	open	device-40	4hU52u6FRdScbqbtv614FQ	1	0	1007	0	389kb	389kb
green	open	device-39	PLt4fyh0S4mJkE0xXPvaFw	1	0	1033	0	394.2kb	394.2kb
green	open	device-38	c9kuxUFQR640XLIJ_6LUA	1	0	1017	0	382.7kb	382.7kb
green	open	device-37	JeowhsbtgQCa6Z_9EySWcxQ	1	0	3724	0	737kb	737kb
green	open	device-36	7uSY4izkQrOkX0XzUgWRYA	1	0	1029	0	400.9kb	400.9kb
green	open	device-35	tRS0tvubTjK5pEOjUDFEUQ	1	0	1029	0	393.3kb	393.3kb
green	open	device-34	IjvFHK24S5GQ9iYH1gT8cA	1	0	1027	0	392.8kb	392.8kb
green	open	device-33	43YNGspOSVuAVo0jm_xk2Q	1	0	1060	0	399.7kb	399.7kb
green	open	device-32	_FTGnHZ_R4WL9C0z3RykNg	1	0	1049	0	418.4kb	418.4kb
green	open	device-31	eczP5eUGQTyOUwNUa9pZWA	1	0	1055	0	406.4kb	406.4kb
green	open	device-30	Gw_OkqqFQ-m9qDrPspAUTg	1	0	1077	0	402.9kb	402.9kb
green	open	device-3	lwrPdEvuT2-LSQJzPPPI4Q	1	0	1013	0	389.8kb	389.8kb
green	open	device-4	2Ux_-SmnR52rMrr1zj3tGQ	1	0	1065	0	400kb	400kb
green	open	device-29	94v19xB9RxxV3rLsR7nMMw	1	0	1026	0	392.7kb	392.7kb

Figure 3 – List of created indices with generated data

Index duplication. The first way to prevent data loss is to create a duplicate index. This method should be considered one of the most possible. This does not provide complete security, but in case of problems with the current index allows you to switch to snapshot quickly. In Figure 4, the execution of the query for duplicating indexes

with data is demonstrated. For web service testing purposes, the Postman desktop application is used. This tool allows querying web resources. It is beneficial when you need to make requests to create, update, and delete documents. A regular browser can be used to view GET requests, as shown in Figure 3.

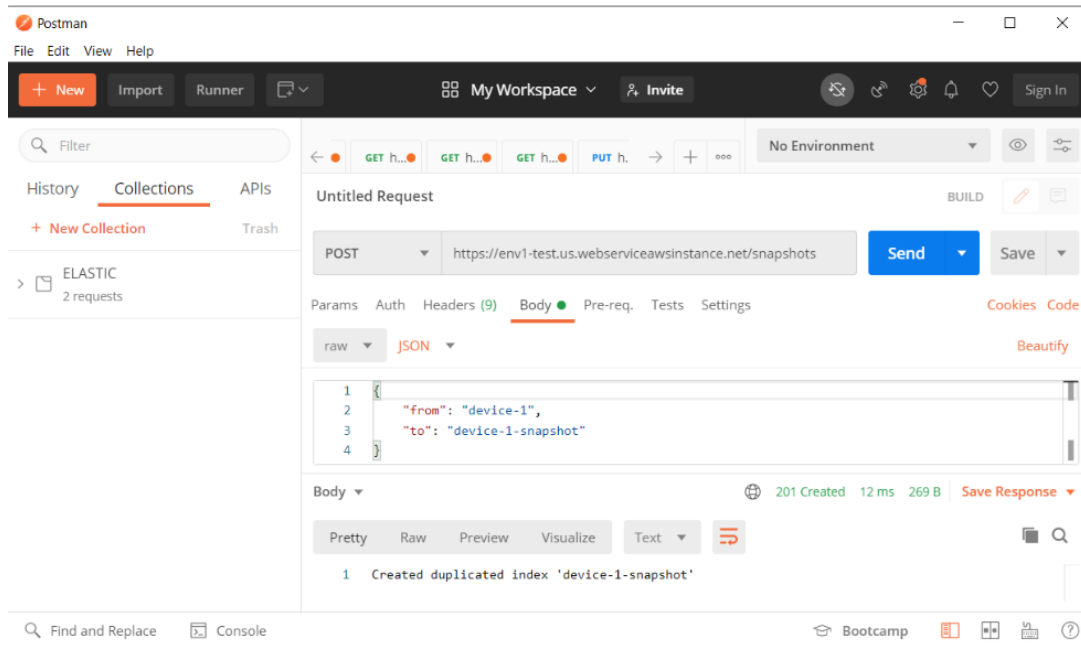


Figure 4 – Executing a query using a web service to create duplicate indexes on the Postman tool

After duplication, the existing Elasticsearch indexes can be checked. Figure 5 shows the dupli-

cated device-1-snapshot index.

health	status	index	uuid	pri	rep	docs.count	docs.deleted	store.size	pri.store.size
green	open	device-19	Syc0bY8V50CQFkYA1M1B0Q	1	0	3699	0	789.7kb	789.7kb
green	open	device-18	cmw8ob47Tzuzf3dyng5Q1w	1	0	3634	0	721kb	721kb
green	open	device-17	1IZFvK-vQ1SZtG-nQQyUg	1	0	3573	0	712.9kb	712.9kb
green	open	device-16	vOKTBohSTFusukyXh1rSPg	1	0	3616	0	721.3kb	721.3kb
green	open	device-15	Wb8em6ikQG6G5A1omChWNg	1	0	3568	0	710.1kb	710.1kb
green	open	device-14	Mt_Shr6grWyp_Gxu8ZHB4g	1	0	3623	0	721.9kb	721.9kb
green	open	device-13	RdvvTeDURtCtMukuwYQRQ	1	0	3615	0	727.6kb	727.6kb
green	open	device-12	ifuMB1CKRt2du5_DOMQC8Q	1	0	3651	0	726.3kb	726.3kb
green	open	device-11	Ux-fL4j1QJmvQms4zGMfFA	1	0	3605	0	725.6kb	725.6kb
green	open	device-10	5M1ZvR23TN6TK1EZ8r4j4g	1	0	3630	0	721.6kb	721.6kb
green	open	device-50	_apuOd41QPKVjvuvvHp6KQ	1	0	3718	0	734.8kb	734.8kb
green	open	device-7	OIHfV_TSSRGkL3GikEHqJA	1	0	3555	0	709.2kb	709.2kb
green	open	device-8	C116YtcVTqunH8hXmmaEFQ	1	0	3607	0	728.7kb	728.7kb
green	open	device-9	KWjTLINrnySSDjgEvJEpw	1	0	3590	0	723.6kb	723.6kb
green	open	device-49	867hJhJrSnCTgyT9AHM6Jg	1	0	3714	0	735.8kb	735.8kb
green	open	device-48	QiZ2aULvRR6Lbyd18Mxtw	1	0	3674	0	728.9kb	728.9kb
green	open	device-47	cLUT21o_sJUSsqv4-nfnZw	1	0	3622	0	720.6kb	720.6kb
green	open	device-46	Vlb1MNR4T-OIsdNaAFmETg	1	0	3591	0	715.8kb	715.8kb
green	open	device-45	XkVearTcStGwGb1R6eYttw	1	0	3703	0	747.7kb	747.7kb
green	open	device-44	i-s81iuiSCGPrFma98HCTw	1	0	3652	0	738kb	738kb
green	open	device-43	D1tumivkRO62N0MSRI6kCA	1	0	3671	0	728.5kb	728.5kb
green	open	device-42	ypqr5jxtQx6oeo-1N4SR6A	1	0	3600	0	725.4kb	725.4kb
green	open	device-41	7tC1ZuTEQAKK-IspaZ1ZiA	1	0	3795	0	770.7kb	770.7kb
green	open	device-40	4HUS2U6FRdSckbqtV614fQ	1	0	3566	0	711.1kb	711.1kb
green	open	device-1-snapshot	hGG_Maem5yqLrFD5z8n9VA	1	0	3658	0	657.4kb	657.4kb
green	open	device-39	PLt4fyh0S4mJkE0xPvaFw	1	0	3619	0	719.9kb	719.9kb
green	open	device-38	c9kuxUFQR64OXLIJ_6LUA	1	0	3505	0	701.1kb	701.1kb
green	open	device-37	JeosbTgQCa6Z_9Ey5wXQ	1	0	3724	0	737.2kb	737.2kb
green	open	device-36	7u5Y4izkQrOkX0ZUgwRyA	1	0	3675	0	728.3kb	728.3kb
green	open	device-35	tR50tVubTjK5pEOjUDFEUQ	1	0	3575	0	713.3kb	713.3kb
green	open	device-34	IjvFHK24S5GQ9iYH1gT8cA	1	0	3637	0	723.3kb	723.3kb
green	open	device-33	43YNGsp0S5VUAv0bjm_xk2Q	1	0	3622	0	722.1kb	722.1kb
green	open	device-32	_fTGNHZ_R4wL9C0z3RykNg	1	0	3637	0	744.5kb	744.5kb
green	open	device-31	eczP5eUGQTy0UwNUa9pZwA	1	0	3736	0	739.3kb	739.3kb
green	open	device-30	Gw_OkqqFQ-m9qDrPspAUTg	1	0	3635	0	725kb	725kb
green	open	device-3	lwrPdEvvT2-LSQJzPPP14Q	1	0	3579	0	712.3kb	712.3kb
green	open	device-4	2Ux_-SmmR52rMrr1zj3tGQ	1	0	3633	0	723.5kb	723.5kb

Figure 5 – Verification list of indices

In general, duplicating data is not the best way to avoid data loss, as an increase in records and memory accompanies it.

The number of indices is not a significant factor in productivity. An important parameter is the number of shards that must be selected correctly.

The cluster should be optimised to avoid impact on performance. The best ratio of the number of bits and replicas can be chosen using performance testing [16]. Amazon CloudWatch can be used for monitoring the results of cluster metrics [8].

Automated and manual snapshots

The second way to recover data is to use automated and manual snapshots on Amazon ES if we do not have a duplicated data set.

Elasticsearch requires repository-s3 plugin to work with an S3 bucket. Therefore, the following operations on the command line must be performed:

- Installing repository-s3: `sudo bin/elasticsearch-plugin install repository-s3`
- Restarting Elasticsearch: `sudo systemctl restart elasticsearch.service`

The next step is to log in to the AWS account and create an identity-based user policy (IAM) with the necessary S3 permissions [8].

The next necessary step is to specify a bucket name to register the S3 repository using the following command:

```
curl --location --request PUT
'http://{domain}/_snapshot/backup_repository_s3'
--header 'Content-Type: application/json' --data-raw '{"type": "s3", "settings": {"bucket": "elastic-slm"}}'
```

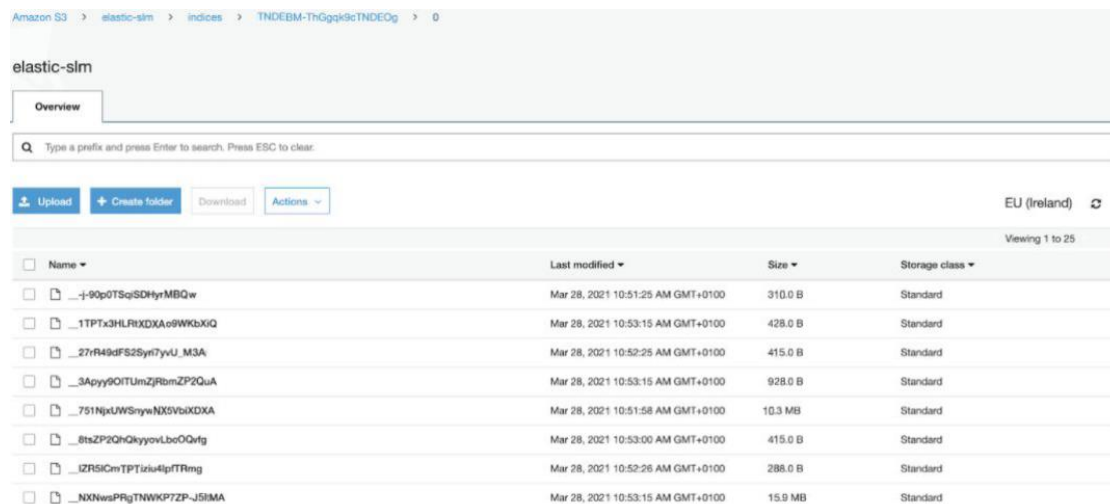
The last setting is defining our Snapshot lifecycle

management (SLM) policy. This means that we will always be able to have a copy of the indexes available, which are generated and deleted at the specified time. In the JSON body, the following fields should be defined: schedule (when the snapshot must be taken into S3 bucket using Cron syntax), name (how to name the snapshot with the current date using date math), repository (where to store the snapshot), config (which indices to include in the snapshot), retention (define the expiration time and the minimum and maximum snapshot count) [17].

To apply these changes, the command should be performed:

```
curl --location --request PUT
'http://{domain}/_slm/policy/backup_policy_daily_s3'
--header 'Content-Type: application/json' --data-raw '{
  "schedule": "0 0 1 2 * * ?", "name": "backup-device-
{now/d}", "repository": "backup_repository_s3", "co
nfig": { "indices": [ "*" ] }, "retention": { "expire_after": "1d",
"min_count": 5, "max_count": 20 } }
```

The web interface for lifecycle management snapshots is shown in Figure 6.



Name	Last modified	Size	Storage class
__j-90p0TSqSDHyMBQw	Mar 28, 2021 10:51:25 AM GMT+0100	310.0 B	Standard
__1TP7x3HLRtXDA09WkxkIQ	Mar 28, 2021 10:53:15 AM GMT+0100	428.0 B	Standard
__27rR49dFS25yr7yu_M3A	Mar 28, 2021 10:52:25 AM GMT+0100	415.0 B	Standard
__3Apyy90ITUmZjRbmZP2QuA	Mar 28, 2021 10:53:15 AM GMT+0100	928.0 B	Standard
__751NpxUWSnywNXSvbiXDXA	Mar 28, 2021 10:51:58 AM GMT+0100	10.3 MB	Standard
__8tsZP2QHkyyovLboOQrtg	Mar 28, 2021 10:53:00 AM GMT+0100	415.0 B	Standard
__IZRSiCmTPTizix4lptFRmg	Mar 28, 2021 10:52:26 AM GMT+0100	288.0 B	Standard
__NXNwsPRgTNWKP7ZP-JSfMA	Mar 28, 2021 10:53:15 AM GMT+0100	15.9 MB	Standard

Figure 6 – Example of snapshot lifecycle management using the web interface on Amazon ES

The AWS command line interface can be used to verify the changed size of our bucket after all previous steps are performed:

```
aws s3 ls s3://elastic-slm --human-readable --recursive --summarize
```

In the same way, indices can be restored using a web interface on Amazon ES.

The restore command must be executed to restore the snapshot after the last step in the AWS command line:

```
curl -XPOST 'elasticsearch-domain-
end-
point/_snapshot/repository/snapshot/_restore'
```

According to Amazon documentation, most automated images are stored in cs-automated storage. If your domain encrypts data at rest, it is stored in the cs-automated-enc repository [11].

One index can be changed to another duplicate index with the same working indices. In ES, bulk index alias API functionality is created to create and remove multiple index aliases in a single API request. Alternatively, an alias can be used as a second name to refer to one or more existing indexes. However, several indexes may contain the same alias, but it is impossible to have the same name as the index. This HTTP request is as follows:

```
POST {ES domain}/_aliases {"ac-
tions":[{"remove":{"index": "device-1", "alias":
"device-1-alias-1"}}, {"add": {"index": "device-
1", "alias": "device-1-alias-2"}}]}
```

REFERENCES

1. Yuzevych, L., Skrynkovskyy, R., Yuzevych, V., Lozovan, V., Pawlowski, G., Yasynskiy, M., & Ogirko, I. (2019). Improving the diagnostics of underground pipelines at oil and gas enterprises based on determining hydrogen exponent (PH) of the soil media applying neural networks. *Eastern-European Journal of Enterprise Technologies*, 4(5), 56–64. doi: [10.15587/1729-4061.2019.174488](https://doi.org/10.15587/1729-4061.2019.174488)
2. Yuzevych, L., Yankovska, L., Sopilnyk, L., Yuzevych, V., Skrynkovskyy, R., Koman, B., Yasynska-Damri, L., Heorhiadi, N., Dzhala, R., & Yasynskiy, M. (2019). Improvement of the toolset for diagnosing underground pipelines of oil and gas enterprises considering changes in internal working pressure. *Eastern-European Journal of Enterprise Technologies*, 6(5), 23–29. doi: [10.15587/1729-4061.2019.184247](https://doi.org/10.15587/1729-4061.2019.184247)
3. Yuzevych, L., Skrynkovskyy, R., & Koman, B. (2017). Development of information support of quality management of underground pipelines. *EUREKA: Physics and Engineering*, 4, 49–60. doi: [10.21303/2461-4262.2017.00392](https://doi.org/10.21303/2461-4262.2017.00392)
4. Gormley, C., & Tong, Z. (2015). *Elasticsearch The Definitive Guide: A Distributed Real-time Search and Analytics Engine*. Retrieved from <http://stmarysguntur.com/wp-content/uploads/2019/04/1021302647.pdf>
5. Kononenko, O., Baysal, O., Holmes, R., & Godfrey, M. W. (2014). Mining modern repositories with elasticsearch. *Proceedings of the 11th Working Conference on Mining Software Repositories*, 328–331. doi: [10.1145/2597073.2597091](https://doi.org/10.1145/2597073.2597091)
6. Taylor, R., Ali, M. H., & Varley, I. (2018). Automating the processing of data in research. A proof of concept using elasticsearch. *International Journal of Surgery*, 55, S41. doi: [10.1016/j.ijssu.2018.05.179](https://doi.org/10.1016/j.ijssu.2018.05.179)
7. Elastic. (n. d.). *Bulk API*. Retrieved from <https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-bulk.html>

CONCLUSIONS

Monitoring data on defects in underground metal structural elements is helpful for the timely prevention of hazards that often occur today during the transportation of various products and substances. For the smooth operation of the proposed system, you need to save the data and, in case of loss, quickly restore it. This article describes how to configure these tools for use. Methods of automatic recovery through snapshots in Amazon Web Service or through duplicate indexes in Elasticsearch are considered. Therefore, given the implemented system, we can make the following results:

- Checked the data recovery application from the non-relational database Elasticsearch and the created web service.
- Tested database completion based on a set of test data using Bash.
- Considered the system for possible detection of defects in underground metal elements of structures.

8. Amazon Web Services. (n. d.). *Analysing Text with Amazon Elasticsearch Service and Amazon Comprehend*. Retrieved from <https://aws.amazon.com/solutions/implementations/text-analysis-with-amazon-opensearch-service-and-amazon-comprehend/>
9. AKCA, M. A., Aydoğan, T., & İlkuçar, M. (2016). An Analysis on the Comparison of the Performance and Configuration Features of Big Data Tools Solr and Elasticsearch. *International Journal of Intelligent Systems and Applications in Engineering*, 4(Special Issue-1), 8–12. doi: 10.18201/ijisae.271328
10. Bahls, D., Zapilko, B., & Tochtermann, K. (2013). A Data Restore Model for Reproducibility in Computational Statistics. *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies*, Article No. 13, 1–18. doi: 10.1145/2494188.2494205
11. Amazon Web Services. (2023). *Amazon OpenSearch Service: Developer Guide*. Retrieved from <https://docs.aws.amazon.com/opensearch-service/latest/developerguide/opensearch-service-dg.pdf#managedomains-snapshots>
12. Mane, D., Chitnis, K., & Ojha, N. (2013). The spring framework: an open source java platform for developing robust Java applications. *Journal of Innovative Technology and Exploring Engineering*, 3, 137–143.
13. Thacker, U., Pandey, M., & Rautaray, S. S. (2017). Review of Elasticsearch Performance Varying the Indexing Methods. *Progress in Intelligent Computing Techniques: Theory, Practice, and Applications*, 3–8. doi: 10.1007/978-981-10-3376-6_1
14. Thomas, M. A., & Redmond, R. T. (2009). From the Client-Server Architecture to the Information Service Architecture. *AMCIS 2009 Proceedings*. 115.
15. Wu, W., Kang, R., & Li, Z. (2015). Risk assessment method for cyber security of cyber physical systems. *2015 First International Conference on Reliability Systems Engineering*. doi: 10.1109/icrse.2015.7366430
16. Elastic. (n. d.). *Quantitative Cluster Sizing*. Retrieved from <https://www.elastic.co/elasticon/conf/2016/sf/quantitative-cluster-sizing>
17. Tencent Cloud. (2022). *Elasticsearch Service Best Practices Product Documentation*. Retrieved from https://main.qcloudimg.com/raw/document/intl/product/pdf/845_16596_en.pdf