# **E4H Dashboard Configuration**

## **Summary**

This document details the steps required to deploy an analytics service in the Incident Management deployment to cater to specific Saura-e-Mitra KPI needs. Note that these cannot be achieved via just Indexer configuration since it involves updating an existing document in the ES Index. This same consumer can be updated in the future to cater to other analytics requirements as well.

## **Overview of Steps**

1. Create PHC master list index mapping by logging into the ES user interface.
2. Upload PHC master list data into the above index via a Python script (details below).
3. Set up an ES pipeline to format the timestamp from the input payload.
4. Deploy im-analytics service in the environment. This must be done before user creation in PHCs.
5. Create Kibana read-only user/password combination for administrators via the Kibana UI.
6. Create Kibana dashboard visualizations and create a public URL for access.

## **Architecture Diagram for Analytics**

## 

## **Detailed Steps**

### **Step 1: Create the Index Mapping via ES DevTools Console**

Open Kibana visualization and click on sidebar at top left corner and open dev tools in it

Create an index with the name phc-master-list and use the following mappings:

| "mappings": {  "properties": {  "Data": {  "properties": {  "@timestamp": { "type": "date" },  "accountCreationTime": { "type": "long" },  "block": { "type": "text", "fields": { "keyword": { "type": "keyword", "ignore\_above": 256 }}},  "code": { "type": "keyword" },  "district": { "type": "text", "fields": { "keyword": { "type": "keyword", "ignore\_above": 256 }}},  "geo-point": { "type": "geo\_point" },  "isLive": { "type": "text", "fields": { "keyword": { "type": "keyword", "ignore\_above": 256 }}},  "name": { "type": "text", "fields": { "keyword": { "type": "keyword", "ignore\_above": 256 }}},  "phcType": { "type": "text", "fields": { "keyword": { "type": "keyword", "ignore\_above": 256 }}},  "tenantId": { "type": "text", "fields": { "keyword": { "type": "keyword", "ignore\_above": 256 }}},  "type": { "type": "text", "fields": { "keyword": { "type": "keyword", "ignore\_above": 256 }}}  }  }  } } |
| --- |

### **Step 2: Push Data into ES via Python Script**

We can push data into Elasticsearch in the phc-master-list index using a Python script available [here](https://github.com/Selco-Foundation/e4h-incident-management/tree/phc-data-python-script/utility). This script takes an Excel file as input.

1. Download the python file on your local machine
2. Open the file in code editor
3. Port forward elastic search data pod from the environment.
4. Use the following command kubectl port-forward elasticsearch-data-0 9200:9200 -n backbone-dev
5. Run the python script
6. Enter the excel path, es host, es username, es password and index name as input
7. Data will be pushed

### **Step 3: Create a Pipeline in ES**

Create a pipeline to convert the accountTimeCreation into date format and assign value to the @timestamp field, which will be helpful for aggregating data month-wise for data visualization.

A. Create a pipeline from the DevTools console:

json

Copy code

PUT \_ingest/pipeline/add\_timestamp\_pipeline

{

"description": "Pipeline to add @timestamp field inside data",

"processors": [

{

"script": {

"lang": "painless",

"source": """

long epochMillis = ctx.Data.accountCreationTime;

java.util.Date date = new java.util.Date(epochMillis);

java.text.SimpleDateFormat sdf = new java.text.SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss.SSS'Z'");

sdf.setTimeZone(java.util.TimeZone.getTimeZone("UTC"));

ctx.Data['@timestamp'] = sdf.format(date);

"""

}

}

]

}

B. Apply the pipeline to the index to update the date field:

json

Copy code

POST /phc-master-list/\_update\_by\_query?pipeline=add\_timestamp\_pipeline

{

"query": {

"match\_all": {}

}

}

### **Step 4: Deploy the im-services-analytics Service**

* Code: [GitHub Repository](https://github.com/Selco-Foundation/e4h-incident-management/tree/im-services-analytics/backend/core-services/im-services-analytics)
* Helm Charts: [Helm Repository](https://github.com/Selco-Foundation/DIGIT-DevOps/blob/DIGIT-2.9LTS/deploy-as-code/charts/core-services/im-services-analytics/values.yaml)
* Image ID: docker.io/egovio/im-services-analytics:im-services-analytics-e827df64db-9

**Configuration:**

#### **EGOV\_INFRA\_INDEXER\_HOST**

- name: EGOV\_INFRA\_INDEXER\_HOST

valueFrom:

configMapKeyRef:

name: egov-config

key: es-indexer-host

**Description:** The host URL of the Elasticsearch. Replace egov-config with your ConfigMap name and es-indexer-host with your key, according to your Elasticsearch host URL.

#### **SPRING\_KAFKA\_PRODUCER\_VALUE\_SERIALIZER**

{{- if index .Values "global" "serializers-timezone-in-ist" }}

- name: SPRING\_KAFKA\_PRODUCER\_VALUE\_SERIALIZER

value: org.egov.tracer.kafka.serializer.ISTTimeZoneJsonSerializer

{{- end }}

**Description:** Serializer for Kafka producer values to ensure timezone is in IST. This setting is conditional based on the global configuration. Adjust based on your global configuration needs.

#### **JAVA\_OPTS**

- name: JAVA\_OPTS

value: {{ index .Values "heap" | quote }}

**Description:** Java options for setting heap memory. Adjust the heap value ( heap: "-Xmx256m -Xms256m" ) in your Helm chart to specify the desired memory settings.

#### **SERVER\_PORT**

- name: SERVER\_PORT

value: "8080"

**Description:** The port on which the application server will run. Change "8080" to your desired server port.

#### **SECURITY\_BASIC\_ENABLED**

- name: SECURITY\_BASIC\_ENABLED

value: "false"

**Description:** Enable or disable basic security. Set to "true" or "false" as per your security requirements.

#### **MANAGEMENT\_SECURITY\_ENABLED**

- name: MANAGEMENT\_SECURITY\_ENABLED

value: "false"

**Description:** Enable or disable management security. Set to "true" or "false" as per your management security requirements.

#### **EGOV\_STATELEVEL\_TENANTID**

- name: EGOV\_STATELEVEL\_TENANTID

valueFrom:

configMapKeyRef:

name: egov-config

key: egov-state-level-tenant-id

**Description:** State-level tenant ID for the system. Replace egov-config with your ConfigMap name and egov-state-level-tenant-id with your key, according to your state-level tenant ID.

#### **EGOV\_INDEXER\_ES\_USERNAME**

- name: EGOV\_INDEXER\_ES\_USERNAME

valueFrom:

secretKeyRef:

name: elasticsearch-credentials

key: username

**Description:** Username for Elasticsearch authentication. Replace elasticsearch-credentials with your Secret name and username with your key, according to your Elasticsearch credentials configuration.

#### **EGOV\_INDEXER\_ES\_PASSWORD**

- name: EGOV\_INDEXER\_ES\_PASSWORD

valueFrom:

secretKeyRef:

name: elasticsearch-credentials

key: password

**Description:** Password for Elasticsearch authentication. Replace elasticsearch-credentials with your Secret name and password with your key, according to your Elasticsearch credentials configuration.

#### **KAFKA\_TOPICS\_CONSUMER**

- name: KAFKA\_TOPICS\_CONSUMER

value: "your-kafka-topic"

**Description:** Kafka topic for the consumer to listen to. Replace "your-kafka-topic" with the Kafka topic you want the consumer to listen to.

#### **EGOV\_UPDATE\_INDEX\_PATH**

- name: EGOV\_UPDATE\_INDEX\_PATH

value: "your-index-name/\_update/"

**Description:** Update index path for Elasticsearch. Replace "your-index-name" with the index name you want to use.

**Deployment Details:**

1. **Add the Helm Charts:**
   * Place the im-services-analytics Helm chart in the core-services directory.
2. **Update the Common Helmfile Configuration:**
   * Update the coreservices-helmfile.yaml to include the configuration for im-services-analytics.

Add this configuration in the  **coreservices-helmfile.yaml:**- name: im-services-analytics

installed: true

chart: ./im-services-analytics

version: 0.1.0

<<: \*default

values:

- ./im-services-analytics/values.yaml

### **Commands to Deploy the Services:**

1. **Export the Kubeconfig file:**  
   export KUBECONFIG=/path/to/your/kubeconfig
2. **Verify the configuration:**helmfile -f '/path/to/your/coreservices-helmfile.yaml' diff  
   This command shows the changes which you will deploy.
3. **Deploy using Helmfile:**  
   helmfile -f '/path/to/your/coreservices-helmfile.yaml' apply

## **Testing the Deployment**

### **Index Level Testing**

1. Create users via HRMS or via Postman scripts.
2. Check the PHC master list index and verify if the isLive flag is true for the PHCs for which users were just created in Step 1.

### **Functional Testing**

1. Check if the dashboard can be accessed via the Kibana read-only user credentials.
2. Check if numbers are being updated live as more centers get rolled out.

### **Data Visualization and Dashboards**

1. Go to Kibana.
2. Open the sidebar in kibana visualization
3. Click on **Stack Management** under the **Management** section.
4. In the **Stack Management** page, click on **Data Views**
5. Enter the name of the index pattern that matches the indices you want to explore
6. For our use case we will be using two indexes phc-master-list and im-services
7. First create two separate data views for both index and After that under timestamp field section select @timetsamp field to do time related analytics
8. After this create another data view with im-service index and this time under timestamp field select i don't want to use timestamp field
9. Save data views to kibana
10. Create visualizations with the following data views.
11. Create a dashboard and publish it.

A sample dashboard is already created under the dashboard section with the name selco-dashboard:

[Selco Dashboard](https://e4h-dev.selcofoundation.org/kibana/app/dashboards#/view/4b2f1950-2ede-11ef-a57d-d987998e399e?_g=(filters:!(),refreshInterval:(pause:!t,value:60000),time:(from:now-6y,to:now%2B6y)))

## **Importing and Exporting Dashboards Across Different Elasticsearch Environments**

### **Prerequisites**

Ensure that the index names, data views, and all mappings are consistent across the environments to avoid any issues during the import and export process.

### **Steps to Export a Dashboard**

1. **Open the Navigation Menu**:
   * Log into your Kibana instance.
   * Open the main navigation menu.
2. **Click on Stack Management**:
   * Navigate to the **Management** section.
   * Select **Stack Management**.
3. **Open Saved Objects**:
   * In the Stack Management section, find and click on **Saved Objects**.
4. **Select the Dashboard to Export**:
   * In the Saved Objects page, locate and select the dashboard you want to export.
5. **Click on the Export Button**:
   * After selecting the desired dashboard, click the **Export** button.
   * The dashboard will be exported as a .ndjson file and downloaded to your local machine.

### **Steps to Import a Dashboard**

1. **Open the Navigation Menu**:
   * Log into the Kibana instance of the target environment.
   * Open the main navigation menu.
2. **Click on Stack Management**:
   * Navigate to the **Management** section.
   * Select **Stack Management**.
3. **Open Saved Objects**:
   * In the Stack Management section, find and click on **Saved Objects**.
4. **Click on the Import Button**:
   * In the Saved Objects page, click the **Import** button.
5. **Import the Exported File**:
   * Select the exported .ndjson file from your local machine.
   * Click **Import** to import the dashboard into the new environment.