Ministry of Housing and Urban Affairs
Government of India

Ministry of Housing and Urban Affairs
Government of India

National Urban Digital Mission

Building cities that work for people

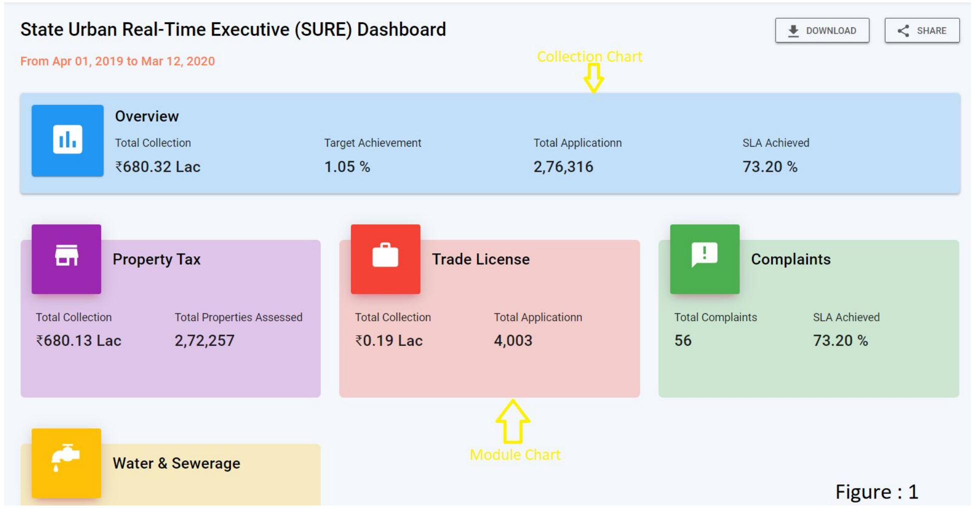
DSS-Dashboard

Outline

1. DSS-Overview
2. DSS-Charts
3. Configuring different charts
4. Indexing the data to elastic search for DSS

DSS-Overview

A decision support system (DSS) is a composite tool that collects, organizes and analyzes business data to facilitate quality decision-making for management, operations and planning.



State Urban Real-Time Executive (SURE) Dashboard

From Apr 01, 2019 to Mar 12, 2020

Overview

Total Collection	Target Achievement	Total Applicationn	SLA Achieved
₹680.32 Lac	1.05 %	2,76,316	73.20 %

Property Tax

Total Collection	Total Properties Assessed
₹680.13 Lac	2,72,257

Trade License

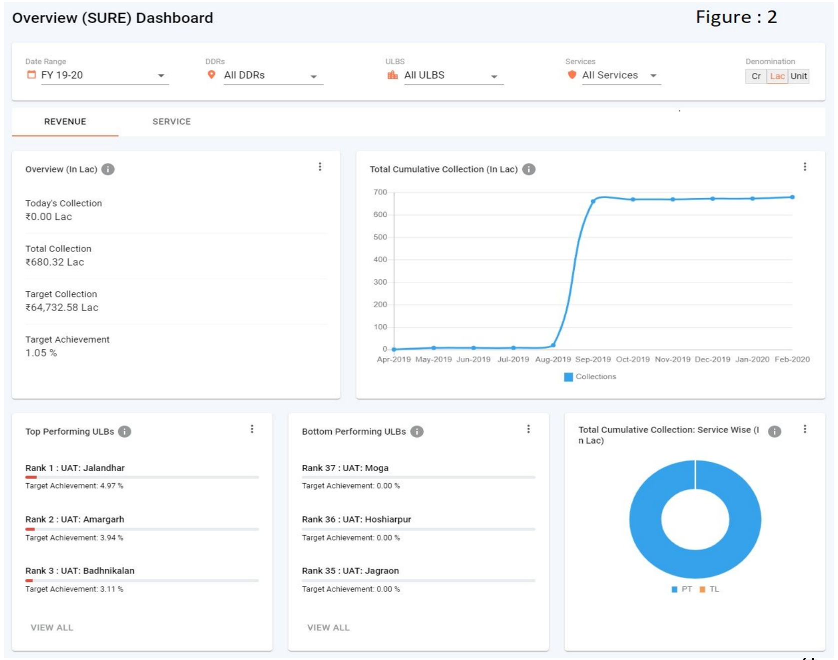
Total Collection	Total Applicationn
₹0.19 Lac	4,003

Complaints

Total Complaints	SLA Achieved
56	73.20 %

Water & Sewerage

Figure : 1



Overview (SURE) Dashboard

Figure : 2

Date Range: FY 19-20 | DDRs: All DDRs | ULBS: All ULBS | Services: All Services | Denomination: Lac

Overview (In Lac)

Today's Collection
₹0.00 Lac

Total Collection
₹680.32 Lac

Target Collection
₹64,732.58 Lac

Target Achievement
1.05 %

Total Cumulative Collection (In Lac)

Top Performing ULBs

Rank 1 : UAT: Jalandhar
Target Achievement: 4.97 %

Rank 2 : UAT: Amargarh
Target Achievement: 3.94 %

Rank 3 : UAT: Badhnikalan
Target Achievement: 3.11 %

Bottom Performing ULBs

Rank 37 : UAT: Moga
Target Achievement: 0.00 %

Rank 36 : UAT: Hoshiarpur
Target Achievement: 0.00 %

Rank 35 : UAT: Jagraon
Target Achievement: 0.00 %

Total Cumulative Collection: Service Wise (In Lac)

Ministry of Housing and Urban Affairs
Government of India

DSS-Charts

i. PIE CHART
ii. LINE CHART
iii. BAR CHART
iv. HORIZONTAL BAR CHART
v. TABLE CHART

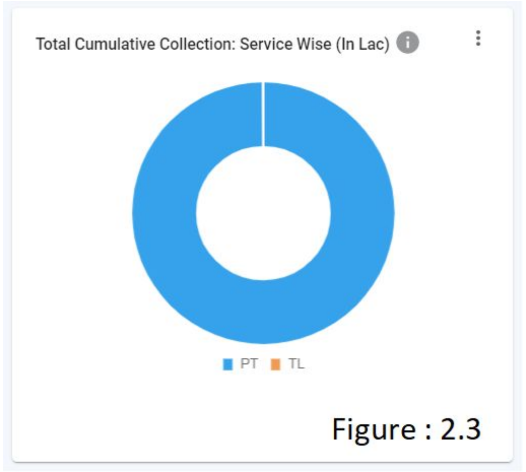
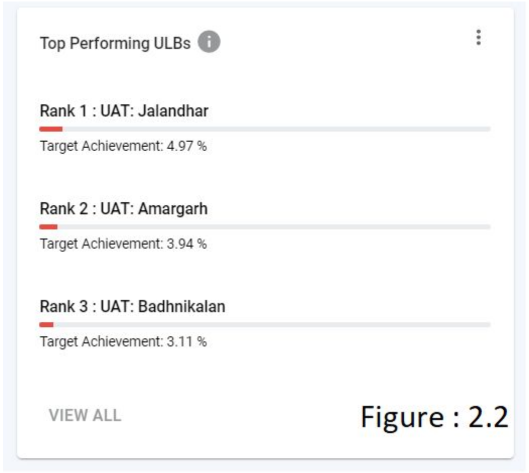


Figure : 2.3


Top Performing ULBs

Rank 1 : UAT: Jalandhar
Target Achievement: 4.97 %

Rank 2 : UAT: Amargarh
Target Achievement: 3.94 %

Rank 3 : UAT: Badhnikalan
Target Achievement: 3.11 %

VIEW ALL

Figure : 2.2

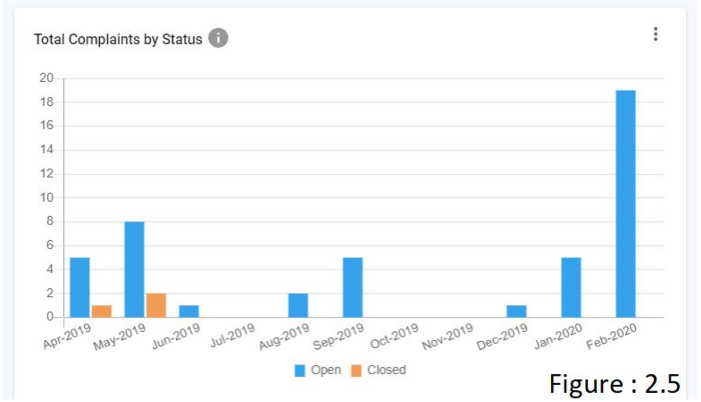


Figure : 2.5

Ministry of Housing and Urban Affairs
Government of India

DSS-Charts

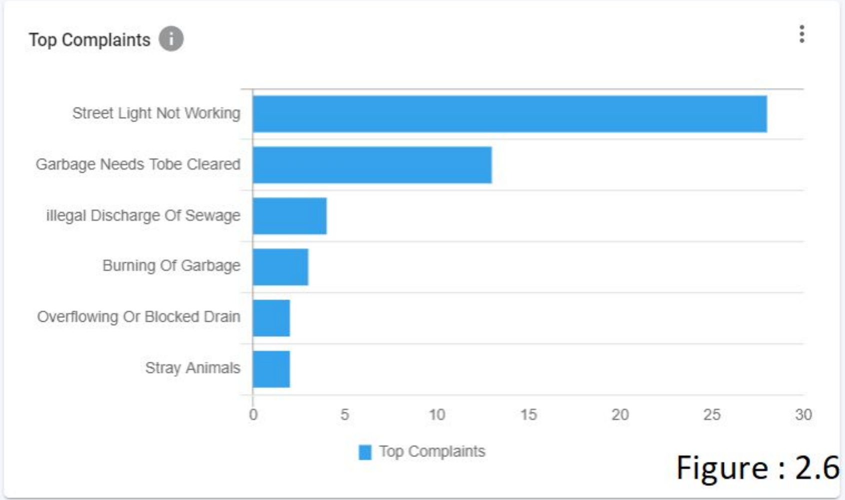
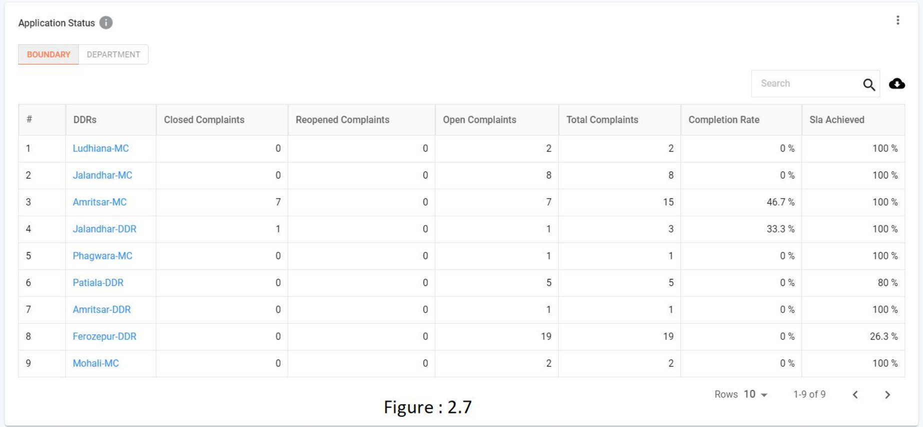


Figure : 2.6



#	DDRs	Closed Complaints	Reopened Complaints	Open Complaints	Total Complaints	Completion Rate	Sla Achieved
1	Ludhiana-MC	0	0	2	2	0 %	100 %
2	Jalandhar-MC	0	0	8	8	0 %	100 %
3	Amritsar-MC	7	0	7	15	46.7 %	100 %
4	Jalandhar-DDR	1	0	1	3	33.3 %	100 %
5	Phagwara-MC	0	0	1	1	0 %	100 %
6	Patiala-DDR	0	0	5	5	0 %	80 %
7	Amritsar-DDR	0	0	1	1	0 %	100 %
8	Ferozepur-DDR	0	0	19	19	0 %	26.3 %
9	Mohali-MC	0	0	2	2	0 %	100 %

Rows 10 1-9 of 9

Figure : 2.7

Configuring different charts

https://github.com/nugp-digit/nugp-configs/blob/master/configs/egov-dss-dashboards/dashboard-analytics/ChartApiConfig.json

Parameter Name	Description
Key (e.g: totalApplication)	This is the key which will be used by client application to indicate which visualization is needed for display.
chartName	The name of the Chart which has to be used as a label on the Dashboard.
queries	The queries of aggregation which are to be used to fetch out the right data in the right aggregated format are configured here.
queries.module	The module/domain level, on which the query should be applied on. Property Tax is PT, Trade License is TL. If the query is applied across all modules, the module has to be defined as COMMON
aggregationPaths	All the queries will be having Aggregation names in it. In order to fetch the value out of each Aggregation Responses, the name of the aggregation in the query will be an easy bet. These aggregation paths will have the names of Aggregation in it.

queries.indexName	The name of the index upon which the query has to be executed is configured here.
queries.aggrQuery	The aggregation query in itself is added here. Based on the Module and the Index name specified, this query is attached to the filter part of the complete search request and then executed against that index
queries.requestQueryMap	Client Request would carry certain fields which are to be filtered. In order to map the parameters of the request to the parameters of the ElasticSearch Document, this mapping is maintained
action	Some of the visualizations are not just aggregation on data source. There might be some cases where we have to do a post aggregation computation. For Example, in the case of Top Performing ULBs, the Target and Total Collection is obtained and then the percentage is calculated.
documentType	The type of document upon which the query has to be executed is defined here.
drillChart	If there is a drill down on the visualization, then the code of the Drill Down Visualization is added here.

queries.dateRefField	Each of these modules have separate indexes. And all of them have their own date fields. When there is a date filter applied against these visualizations, each of them has to apply it against their own date reference fields. In order to specify what is the date field in which index, we have this configured in this configuration parameter.
chartType	This field defines as what is the type of chart / visualization that this data should be used to represent.
valueType	In order to represent them and differentiate the numbers from amount from percentage, this field is used to indicate the type of value that this visualization will be sending.

Ministry of Housing and Urban Affairs
Government of India

Configuring Master board

Parameter Name	**Description**
name	Name of the Dashboard which has to be displayed as Page Heading
id	Unique Identifier of the Dashboard which should be used later for Querying each of these Visualizations
isActive	Active Indicator which can be used to quickly disable a dashboard if required.
style	Style of the Dashboard. Whether it should be a linear one or tabbed one.
visualizations	The list of visualizations which are to be displayed in the Dashboard are listed out here.
visualizations.row	The row identifier for each Visualization are mentioned here

visualizations.vizArray.dimensions	Each of these group of charts are given a dimension based on which they are placed in a specific row in a dashboard
visualizations.vizArray.charts	The list of individual charts inside a Visualization Group is maintained in this array list

Ministry of Housing and Urban Affairs
Government of India

Configuring Indexer

https://github.com/UPYOG-UPYOG/UPYOG-configs/blob/master/configs/egov-indexer/egov-fsm.yml

Variable Name	Description
topic	Topic on which the data is to be received to activate this particular configuration.
serviceName	Name of the module to which this configuration belongs.
outJsonPath	JSONPath of the field of the index json.
name	Index name on the elasticsearch. (Index will be created if it doesn't exist with this name.
inJsonPath	JSONPath of the field from the input.
indexMapping	A skeleton/mapping of the JSON that is to be indexed.
externalUriMapping	Contains a list of configurations. Each configuration contains keys to identify the field of the input JSON that are to be enriched using APIs from the external services. The configuration for those APIs also is a part of this.

Ministry of Housing and Urban Affairs
Government of India

Indexing

For DSS we are using elasticsearch for getting the data .
So we need to follow few steps to push the data to elasticsearch for dss .

Connect playground:

```
kubectl exec -it <playground> -n playground bash --kubeconfig <kube config path>
```

Check Indices and Alias:

```
curl -X GET 'http://elasticsearch-data-v1.es-cluster:9200/_cat/indices?v'
curl -X GET 'http://elasticsearch-data-v1.es-cluster:9200/_cat/aliases?v'
```

If index and aliases are not present -
Create Indices and Alias:

```
curl -XPUT -H "Content-Type: application/json"  http://elasticsearch-data-v1.es-cluster:9200/egov-dss-ingest-enriched?pretty

curl -XPOST "http://elasticsearch-data-v1.es-cluster:9200/_aliases" -H 'Content-Type: application/json' -d'
{
  "actions" : [
     { "add" : { "index" : "egov-dss-ingest-enriched", "alias" : "dss-collection_v2" } }
  ]
}'
```

Ministry of Housing and Urban Affairs
Government of India

Create connector

We need to create kafka connector so that it'll pipeline the data to elastic search for indexing

```
curl -X POST \
http://kafka-connect.kafka-cluster:8083/connectors/ \
-H 'Content-Type: application/json' \
-H 'Cookie: SESSIONID=f1349448-761e-4ebc-a8bb-f6799e756185' \
-H 'Postman-Token: adabf0e8-0599-4ac9-a591-920586ff4d50' \
-H 'cache-control: no-cache' \
-d '{
"name": "<connector-name>",
"config": {
"connector.class": "io.confluent.connect.elasticsearch.ElasticsearchSinkConnector",
"connection.url": "http://elasticsearch-data-v1.es-cluster:9200",
"type.name": "general",
"topics": "<index-name>",
"key.ignore": "false",
"schema.ignore": true,
"value.converter.schemas.enable": false,
"key.converter": "org.apache.kafka.connect.storage.StringConverter",
"value.converter": "org.apache.kafka.connect.json.JsonConverter",
"transforms": "TopicNameRouter",
"transforms.TopicNameRouter.type": "org.apache.kafka.connect.transforms.RegexRouter",
"transforms.TopicNameRouter.regex": ".*",
"transforms.TopicNameRouter.replacement": "<index-name>",
"batch.size": 10,
"max.buffered.records": 500,
"flush.timeout.ms": 600000,
"retry.backoff.ms": 5000,
"read.timout.ms": 10000,
"linger.ms": 100,
"max.in.flight.requests": 2,
"errors.log.enable": true,
"errors.deadletterqueue.topic.name": "<index-name>-es-failed",
"tasks.max": 1
}
}'
```

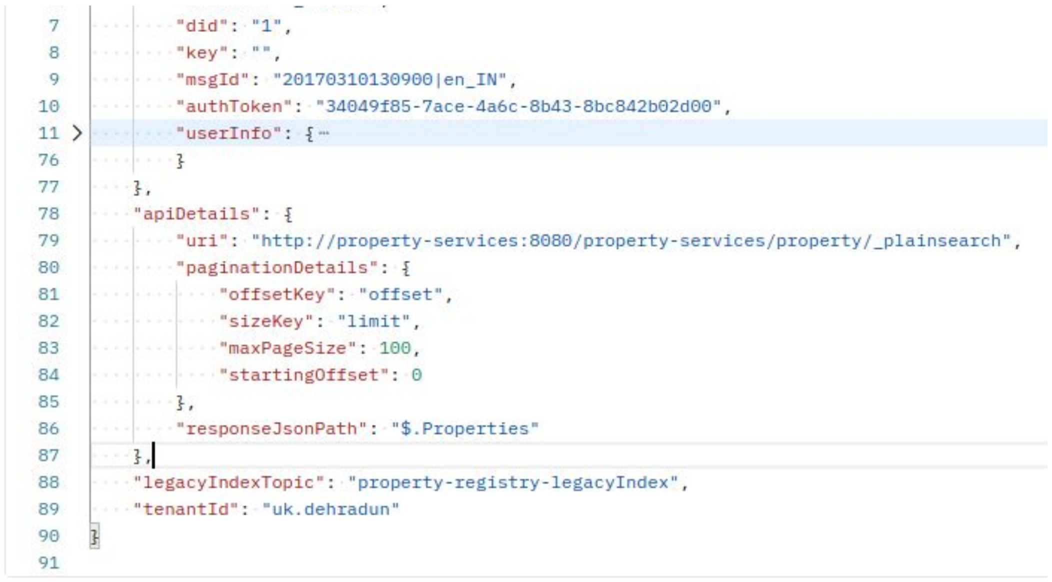
Connector name ->unique name of the connector
Index name -> Index for which you're creating this connector

Restart Indexer
After creating kafka connector we need to restart indexer

Port forward the indexer pod and run the postman script for indexing the data
Here in the request body we need to change the api details according to the module for which \
we want to index

Postman collection->

https://www.getpostman.com/collections/eee60aebc90a04f6591b



```
            "did": "1",
            "key": "",
            "msgId": "20170310130900|en_IN",
            "authToken": "34049f85-7ace-4a6c-8b43-8bc842b02d00",
            "userInfo": {⋯
            }
        },
        "apiDetails": {
            "uri": "http://property-services:8080/property-services/property/_plainsearch",
            "paginationDetails": {
                "offsetKey": "offset",
                "sizeKey": "limit",
                "maxPageSize": 100,
                "startingOffset": 0
            },
            "responseJsonPath": "$.Properties"
        },
        "legacyIndexTopic": "property-registry-legacyIndex",
        "tenantId": "uk.dehradun"
}
```

Thank You

"UPYOGal Transformation is more about humans than UPYOGal"