Table of Contents

1 Introduction

This design document is to explain how to implement the requirements for Elasticsearch Application,

- Describes about process of FSCrawler's job reading PDF files and ingests into Elasticsearch.
- Describes about process of Logstash's job reading meta data from claims database and insert/ingest them into ElasticSearch.
- Applying searches on templates (the ingested pdfs' details).

2 Purpose

The purpose of the document is to outline the detail design for applying Search on templates, which can get from OCR Output Folder.

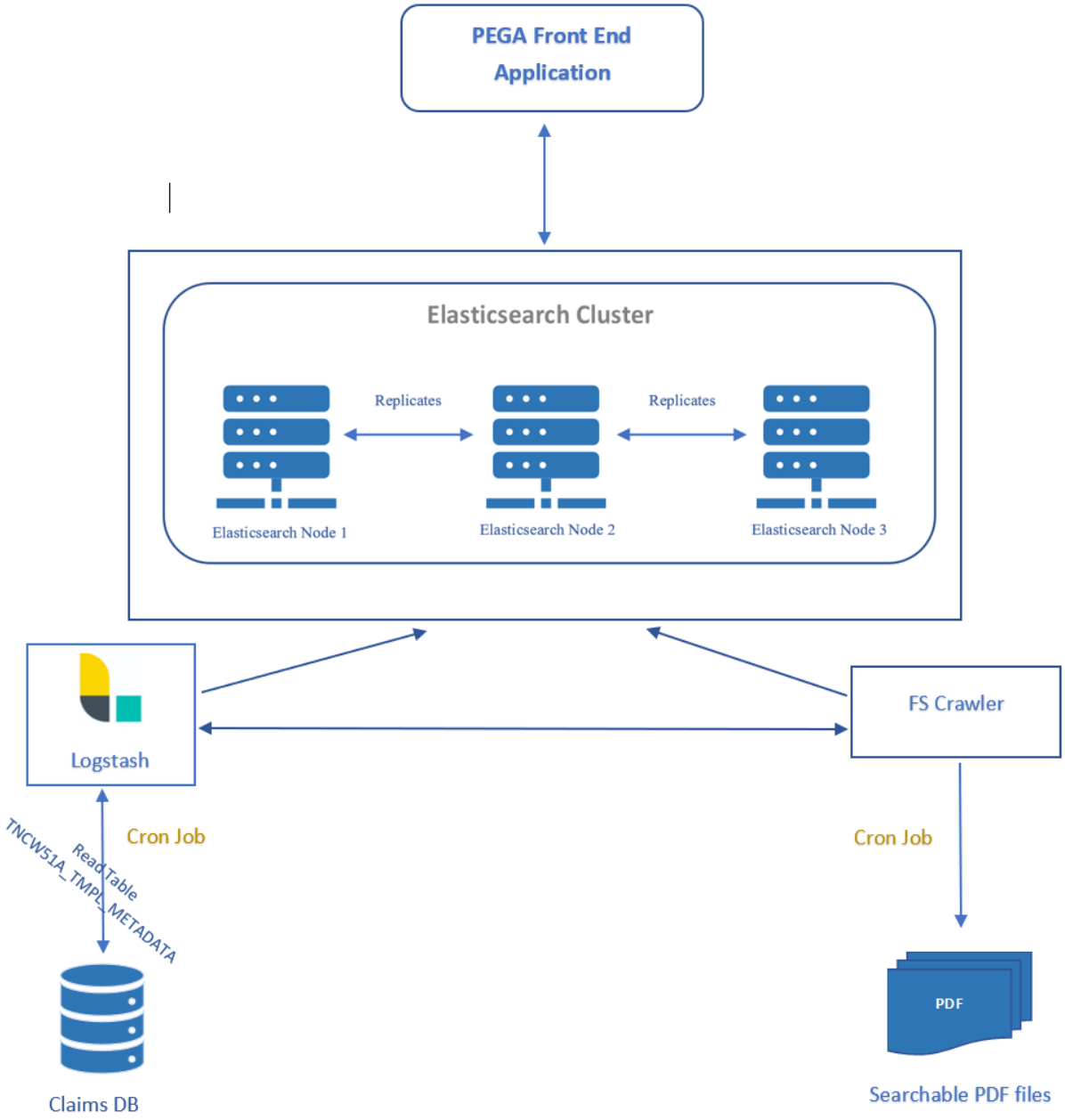
3 Technical Scope

JDK 1.8
Elastic Search 6.8.14
Logstash 6.8.14
Kibana 6.8.14
FSCrawler 2.4

4 Prerequisites, Dependencies and Assumptions

- PDF files will be available in OCR Output Folder and the files will be get placed by Captiva's OCR process.
- Template meta data will be getting from *TMPL_METADATA* table.
- PEGA system will be having UI to construct and send searches' rules and data as request to Elasticsearch.
- Elasticsearch will be sending back response as json objects.

4.1 Architecture Diagram



1. First, FS Crawler will read PDF files from OCR output folder (<servername>\CognitiveClaims\Export) and will store/ingest into 'templatesindex' Index in Elasticsearch.

2. Then, Logstash will fetch template data's meta Data from TMPL_METADATA table of Claims Database and will transfer the same to 'templatesindex' Index in Elasticsearch.

3. Search operation will be performed from PEGA front end application and the search hits Elastic Search through http rest service with the search string given by user.

4.2 Implementation Details

4.2.1 Process implementation

1. The contents in the index(templatesindex) will be cleared/deleted all data before every load.

Whenever new Pdf files received in the input folder(*through MoveIT job, OCI Export folders' files would move to Input folder in elastic server*), the system will process to load new pdf contents into the index, before that, all existing contents needs to be removed in the index

2. For FSCrawler job, actual file name will be renamed to have only template id, then ingest into Elastic.

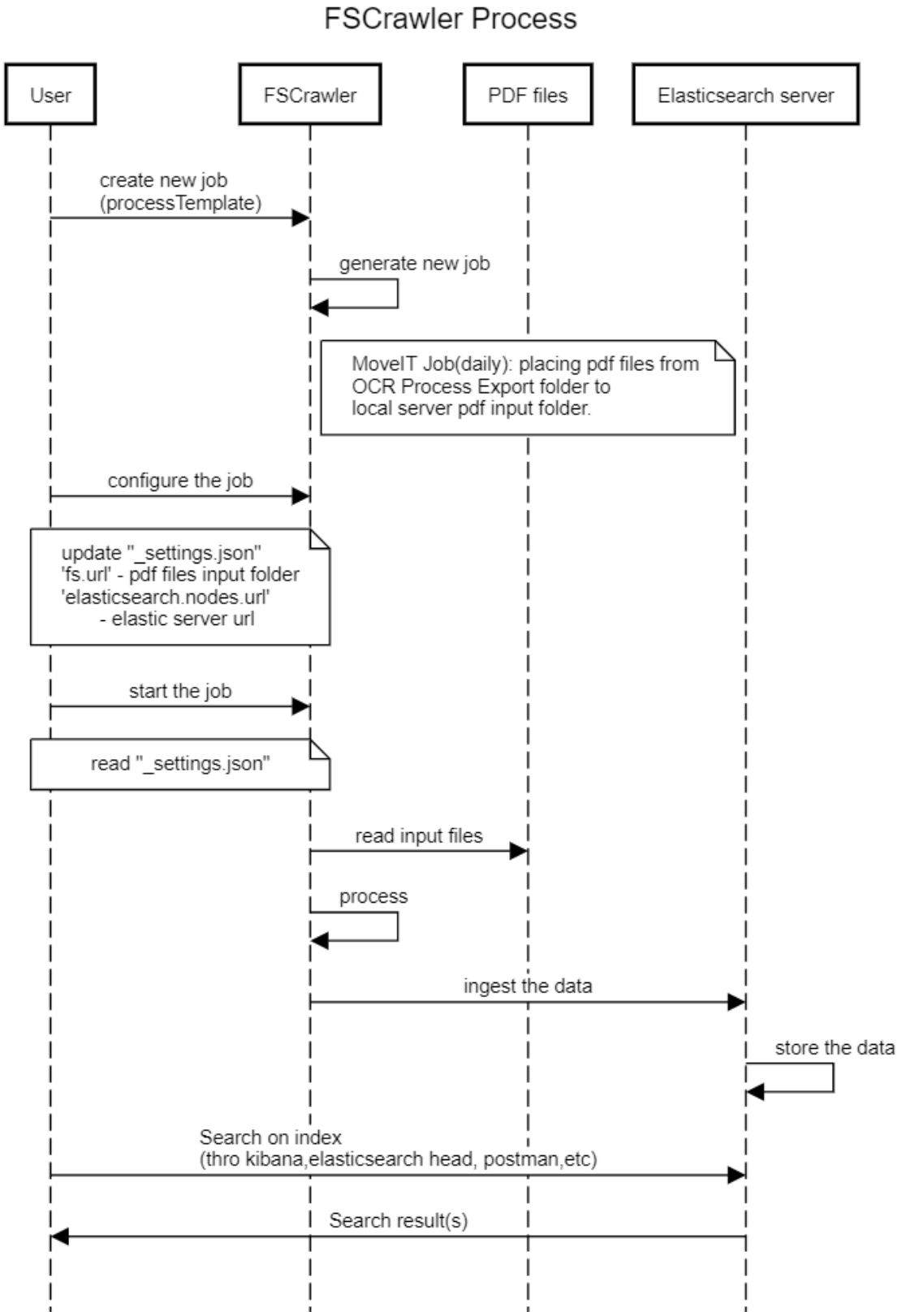
For example, if a pdf file name is

'*FRMTMPL_20210421195910_**4df1c8c7-a3a1-4ee4-bb25-ada3bd7d9a59**.pdf*', then it has to be renamed as '***4df1c8c7-a3a1-4ee4-bb25-ada3bd7d9a59***'.

3. Data from TMPL_METADATA table should be ingested into the same index(templatesindex), not separate two indices.

4. FSCrawler job will process first, then logstash for Templates' metadata will process.
5. So that templates' meta data and pdf contents will be in same record with respective template it.

4.2.2 Sequence diagram FSCrawler process



4.2.3 FSCrawler Process in detail

- Create new Index for pdf search, we need to run the following command in command prompt:
  - ***bin\fscrawler --config_dir ./processTemplates***
  - This would create processTemplates index under forms folder in FSCrawler.
  - Then, we need to update the configuration file ***_settings.json*** *(folder : processTemplates).*
  - The following attributes we need to change based on our requirement.
  - Attributes marked in yellow below :
    - Name: index name
    - url : Your PDF input folder name(based the server operating system path needs to be placed).
    - Elastisearch.host and port : Elasticsearch installed server ip/dns name and port number.
    - Rest.host and port : FSCrawler installed server ip/dns name and port number to make rest call to FSCrawler indices.

```
{
  "name" : "processTemplates",
  "fs" : {
    "url" : "C:\\ELKStack\\data\\CognitiveClaims\\Export",
    "update_rate" : "15m",
    "includes" : [ "*.*" ],
    "excludes" : [ "*/~*" ],
    "json_support" : false,
    "filename_as_id" : true,
    "add_filesize" : true,
    "remove_deleted" : true,
    "add_as_inner_object" : false,
    "store_source" : false,
    "index_content" : true,
    "attributes_support" : false,
    "raw_metadata" : true,
    "xml_support" : false,
    "index_folders" : true,
    "lang_detect" : false,
    "continue_on_error" : false,
    "pdf_ocr" : true,
    "ocr" : {
      "language" : "eng"
```

```
  }
 },
 "elasticsearch" : {
  "index" : " templatesindex",
   "nodes" : [ {
     "host" : " http://d0816v.server.uat",
     "port" : 30007,
     "scheme" : "HTTP",
     "username" : "elastic",
     "password" : "${ES_PASSWORD}"
   } ],
   "bulk_size" : 100,
   "flush_interval" : "5s",
   "byte_size" : "10mb"
 },
 "rest" : {
   "scheme" : "HTTP",
   "host" : "127.0.0.1",
   "port" : 8080,
   "endpoint" : "fscrawler"
 }
}
```

- Once changes will be done, start the fscrawler server *(Note: before this you should have started ElasticSearch server and Kibana server-if you will do query from Kibana's Dev tool)*.
- **bin\fscrawler --config_dir ./processTemplates --loop 1**

- After run above command, the new 'processTemplates' index will be created in FSCrawler and fscrawler will be read configured 'OCR output folder' as input to the job, then will ingest pdf files' details into 'processTemplates' index.
- There will be a scheduled Cronjob, which runs on daily night at a specified time to run above command.
- After the process done, move the PDF files into an Archive Folder.

4.2.4 Process in Logstash

- After FSCrawler process done, another job(scheduled Cronjob) will be executed to trigger Logstash to pull data from TMPL_METADATA table.
- Logstash will have configuration like where to connect for DB2 and from which table need to pull the data. Once the job triggered, Logstash will

pull the data and send them into Elasticsearch on the same ‘templatesindex’ index.

```
input {
    jdbc {
      jdbc_driver_library => "/usr/share/logstash/lib/jdbc/db2/db2jcc.jar"
      jdbc_driver_class => "com.ibm.db2.jcc.DB2Driver"
      jdbc_connection_string =>
"jdbc:db2://m35svr.mega.us.data.com:6510/M2T1:retrieveMessagesFromServerOnGet
Message=true;"
      jdbc_user => "${DB2_UserName}"
      jdbc_password => "${DB2_Password}"
      statement => "SELECT * FROM DATADBA8.TMPL_METADATA"
    }
  }
  filter {
  }
  output {
    elasticsearch {
      hosts => ["http://d0816v.server.uat:30007"]
      user => "elastic"
      password => "${ES_PASSWORD}"
      index => "templatesindex"
      document_id => "%{doc_tmpl_id}"
      action => "update"
      doc_as_upsert => true
    }
  }
```

logstash_dev2_db.conf

Please place this file under folder: /logstash-6.8.14/config

To run the configuration file:

Open a command prompt and move to /logstash-6.8.14, then run this command.

```
bin/logstash -f logstash_dev2_db.conf
```

4.2.5 Searching on Elasticsearch for templates

There are in multiple combinations search can be applied, Case Summary, Form Letter, State Form Internal Form, Form Number (or) Form Name.

All searches can be done through REST (or) Elasticsearch-Head browser plugin.

(i) Search for all formnumber matching the value

```
GET templatesindex/_search
{
  "query":{
    "match":{
            "formNumber":"2586032432"
    }
  }
}
```

(ii) Search for all formnumber starts with given value

```
GET templatesindex/_search
{
  "query":{
    "prefix":{
            "formNumber":"5022632605"
    }
  }
}
```

(iii) Search in multiple fields for given value

```
GET templatesindex/_search
{
  "query":{
      "multi_match" : {
       "query": "5022632605",
       "fields": [ "formNumber", "caseSummary" ]
     }
  }
}
```