

1/15/2019

TF-AMD LogSearch System

HANDOVER DOCUMENTATION

XIE, SHAE

Contents

Development Tools.....	3
Program	4
Back-End	4
1. Elasticsearch	4
2. Logstash.....	6
WebServer.....	8
1. Nodejs	8
2. Ajax + JQuery	9
Front-End.....	10
1. Kibana.....	10
2. Plotly	12
3. UI.....	14
Study Guide	17
Back-End	17
1. Elasticsearch	17
2. Logstash.....	19
WebServer.....	20
1. Nodejs	20
2. Ajax.....	21
3. JQuery	22
Front-End.....	22
1. Kibana.....	22
2. Plotly	23
Program Language	24
Configuration.....	24
File Structure	24

Deploy	26
Open ES Server	27
Open Web Server	27
Data Update	28
Update Text Files	28
Update Database	29
Future Challenge.....	31
PDF Functionality.....	31
More Diagrams	31
Search Accuracy.....	31
Database Presentation.....	32

Development Tools

Three parts of the whole system:



Program

This section introduces technical details and explains the code.

Back-End

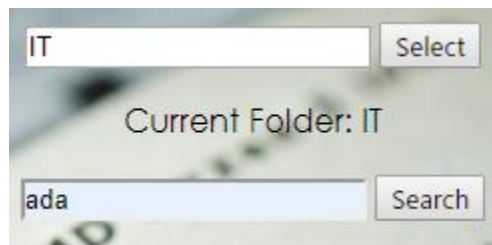
1. Elasticsearch

The most operation in elasticsearch is query according to the given keywords.

File:/stage3 v3 /server searchengine v3.js

There are three different ways to search:

- a. Target Folder : text files only



```
var index = "logstash-db_log-2019.1.3"; // the recent ES database
var results_number = 100;

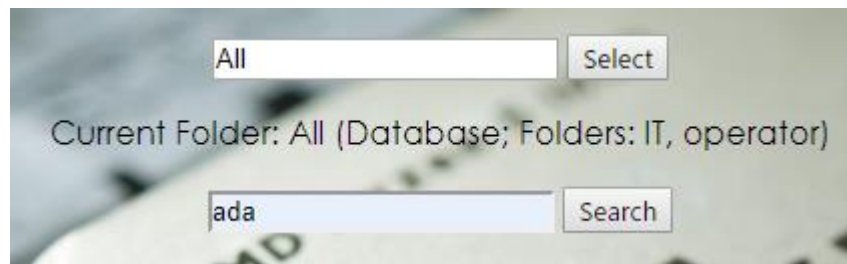
index: index,      // the ES database name
type: 'doc',
size: results_number, // number of return result
body:{
  query:{          // query body
    bool:{
      must:[
        {match:{type:"txt"}},           // search in all test files
        {match_phrase:{message: keyword}}, // search keyword as whole phrase
        {match:{log_folder: folder}}     // match the keyword
      ]
    }
  },
  aggs: {          // group the query result
    type: {
      terms: {
        "field": "type.keyword"
      },
      aggs: {
        folder: {          // group by folder firstly
          terms: {
            "field": "log_folder.keyword"
          },
          aggs: {
            log: {        // group by log name then
              terms: {
```

```

        "field": "log_name.keyword"
    }
    }
    }
    }
    },
    _source: ["log_time", "log_date", "log_name", "message", "log_folder", "type"]
    // only show partial attributes
}

```

b. Search in both database and all files: all data

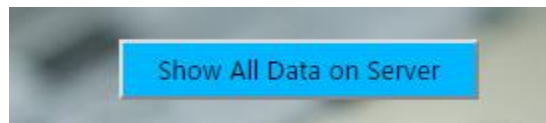


```

index: index,
type: 'doc',
size: results_number,
body: {
  query: {
    bool: {
      should: [
        {match: {db_message: {"query": keyword, "operator": "and"}}}, // database
        {match_phrase: {"message": keyword}} // files
      ]
    }
  },
  aggs: {
    type: {
      terms: {
        "field": "type.keyword"
      },
      aggs: {
        folder: {
          terms: {
            "field": "log_folder.keyword"
          },
          aggs: {
            log: {
              terms: {
                "field": "log_name.keyword"
              }
            }
          }
        }
      }
    }
  }
}

```

c. Show the overall data statics



```
index: index,
type: 'doc',
size: results_number,
body: {
  aggs: {
    type: {
      terms: {
        "field": "type.keyword"
      },
      aggs: {
        folder: {
          terms: {
            "field": "log_folder.keyword"
          },
          aggs: {
            log: {
              terms: {
                "field": "log_name.keyword"
              }
            }
          }
        }
      }
    }
  }
}
```

2. Logstash

Logstash is needed when import and update data, as well as connect to the conventional database and parse text file.

File: /conf/logstash/

a. Import and parse data from text files under folders

```
input {
  file {
    path => "C:/Mulong/logs/operator/480b5c800056afd8-(BRM_PL_TNR-
    BRTMP36).txt"
    start_position => "beginning"
    codec => multiline{
      negate => true
      pattern => "(^|\\[INFO\\]|)(\\d+\\-\\w+\\-\\d+)\\s(\\d+\\.\\d+\\.\\d+)"
      what => "previous"
    }
  }
}
```

Set file path

Define log segment

Differentiate from database

```

    add_field => {"type" => "txt"}
  }
}

filter {
  grok {
    break_on_match => false
    match => {"message"=>
      "(^|\\[INFO\\]\\[\\])(?<log_date>\\d+\\-\\w+\\-\\d+)\\s(?<log_time>\\d+\\.:\\d+\\.:\\d+\\.\\.?\\d*)[\\]\\s\\t]*(?<log_content>\\.*)"
    }
    match => {"path"=>"(?<log_folder>[^\\/]*)\\/(?<log_name>[^\\/]*.\\.(log|txt))"}
  }
}

```

Parse unique log format

b. Connect to database

```

input{
  jdbc{
    jdbc_driver_library => "c:\\Mulong\\jdbc\\ojdbc8.jar"
    jdbc_driver_class => "Java::oracle.jdbc.driver.OracleDriver"
    jdbc_connection_string =>
      "jdbc:oracle:thin:edr_admin/edr_admin32229@//vpngorasvdlstg:1521/svdlqa"
    jdbc_user => "edr_admin"
    jdbc_password => "edr_admin32229"
    statement_filepath => "C:\\Mulong\\git\\github\\Elasticsearch-nodejs-
      UI\\nodejs_project_AMD\\sql\\1.sql"
    type => "todo"
  }
  jdbc{
    jdbc_driver_library => "c:\\Mulong\\jdbc\\ojdbc8.jar"
    jdbc_driver_class => "Java::oracle.jdbc.driver.OracleDriver"
    jdbc_connection_string =>
      "jdbc:oracle:thin:edr_admin/edr_admin32229@//vpngorasvdlstg:1521/svdlqa"
    jdbc_user => "edr_admin"
    jdbc_password => "edr_admin32229"
    statement_filepath => "C:\\Mulong\\git\\github\\Elasticsearch-nodejs-
      UI\\nodejs_project_AMD\\sql\\2.sql"
    type => "userpreferences"
  }
}

```

Connect to the DB server

Content you wanna select

Table name

Adding when import new table

WebServer

1. Nodejs

Used to build the web server and excuse the search query according to input keyword, as well as send back the result to front-end.

File:/stage3 v3 /server router v3.js

Initial page

The first page sent when link to the system.

```
// *** router start ***
app.get('/', function (req, res) {
  res.sendfile(__dirname + '/public/index_v3.html');
});
```

a. Main server: trigger the search engine.

```
app.get('/getkeyword', function (req, res) {
  // get keyword from request
  var search = {
    'folder': req.query.folder,
    'keyword': req.query.keyword,
    'show_all': req.query.show_all
  };
  keycontent = search['keyword'];
  folder = search['folder'];
  console.log("\n\ninput folder: " + folder);
  console.log("Input keyword: " + keycontent);

  // trigger the search engine
  // search by given keywords
  es.elasticSearch(search, function (result) {
    var response = {}; // the final return response
    if(result){
      // return variables
      var disp = {}; // the table on website
      var draw_data = {}; // the data for drawing diagram
      // show all or show details
      if(search['show_all']){
        disp = ui.disp_overview(result, draw_data);
      }
      else{
        var overview = ui.disp_overview(result, draw_data);
        disp = ui.disp_detail(result, keycontent);

        disp['txt'] = overview['txt'] + disp['txt'];
        disp['db'] = overview['db'] + disp['db'];
      }

      // gather the results
      response['disp'] = disp;
      response['draw_data'] = draw_data;
      response['status'] = 1;
      res.setHeader('Content-Type', 'text/html');
      res.end(JSON.stringify(response));
    }
  });
});
```

```

    }
    else{
        response['disp'] = "<h3>No related result found by given keyword in target
folder: " + search['folder'] + '/' + search['keyword'] + "</h3>";
        response['status'] = -1;
        res.end(JSON.stringify(response));
    }
});
});

```

b. Download file from local server

```

app.get('/download', function (req, res) {
    var file = __dirname + '\\logs\\' + req.query.file;
    console.log("Download " + file);
    res.set({
        'Content-Type': 'application/octet-stream',
        'Content-Disposition': 'attachment; filename=' + req.query.file
    });
    fs.createReadStream(file).pipe(res);
});

```

2. Ajax + JQuery

Use to transfer data and collect the return data from the server, as well as defining the on_click function.

File:/stage3 v3/public/index v3.html

```

<script type="application/javascript">
    // ajax data transfer
    $('#show_all').click(function () {
        $.ajax({
            url: '/getkeyword',
            type: 'get',
            data: {
                show_all: "show_all"
            },
            success: function (data) {
                // parse the string data into Json
                var data_json = eval('(' + data + ')');
                // display result
                document.getElementById('folder_name').innerText = 'Current Folder:
All logs and database on server';
                document.getElementById('table_txt').innerHTML =
data_json['disp']['txt'];
                document.getElementById('table_db').innerHTML =
data_json['disp']['db'];

                // change display area
                document.getElementById('rt').style.display = 'block';
                document.getElementById('rt').style.backgroundColor = '#00B7FF';
                document.getElementById('nav_bar').style.display = 'block';
                document.getElementById('nav_data_source').style.display = 'block';
                document.getElementById('search_failed').style.display = 'none';
                document.getElementById('disp_table').style.display = 'block';
                document.getElementById('disp_plot').style.display = 'none';

                // change button status
                document.getElementById('butt_table').className = 'active';
            }
        });
    });

```

```

        document.getElementById('butt_plot').className = '';
        plot(data_json['draw_data']);
    }
})
});

```

Front-End

1. Kibana

Kibana is only a test tool in this project to check the correctness of the ES query statement.

Another important function of Kibana is to set the property of each data field (columns of each table in traditional database), it's called mapping.

File: /conf/mapping

a. Query

```
GET /_cat/indices?v
```

Check all data in current ES database

```

GET /logstash-db_log-2019.1.3/_search
{
  "size": 0,
  "aggs": {
    "dif_type": {
      "terms": {
        "field": "type.keyword"
      },
      "aggs": {
        "log_name": {
          "terms": {
            "field": "log_name.keyword"
          }
        }
      }
    }
  }
}

```

Similar grammar as ES query

Kibana is a great place if you want to inspect your database and test your query statement.

b. **Mapping** *

This is highly important for importing data from traditional Oracle /MySQL database, otherwise the full-text (all filed) search may not be achieved.

Notes:

- i. In the “settings”, the number_of_shards is always 5 to guarantee the performance.
- ii. Logstash can help to reset property of columns in old traditional database automatically if no customized mapping is given, BUT you need to do that by yourself manually if you want to achieve full-text search.
- iii. To achieve full-text search, you need to add a “copy_to” attribute to each field. And copy all the fields to that single field.
- iv. All the field that copy to the same filed required same type (usually text)
- v. If you want to give multiple properties to one field, use “fields” setting
- vi. “Keyword” type is required if you want to do aggregation in this field.

<https://www.elastic.co/guide/en/elasticsearch/reference/current/copy-to.html>

Set the number of shards, 5 usually

```
PUT /dbtest2
{
  "settings": {
    "number_of_shards": 5
  },
  "mappings": {
    "doc": {
      "properties": {
        "id": {
          "type": "text",
          "copy_to": "message",
          "fields": {
            "keyword": {
              "type": "keyword",
              "ignore_above": 256
            }
          }
        },
        "location": {
          "type": "text",
          "copy_to": "message"
        }
      }
    }
  }
}
```

Set the new property for each field

All fields should be copy to somewhere to achieve full-text search

New type of this field (column)

Alternative type of this field

Every column in before database needs a new property

Example of create a new Index (Database in MySQL)

2. Plotly

Plotly is a convenient tool to visualize the result to see the trend and distribution. It is built on JavaScript and the code is embedded in script.

[File:/stage3 v3 /public/index v3.html](#)

Note:

The source code package of plotly is needed before using.

```
<script src="public/plotly-latest.min.js"></script>
```

Import the source code

```
// draw the diagram
// data format sample: ["log1":10, "log2":10, "logn":16]
function plot(data) {
  var div = document.getElementById('disp_plot');
  var x = [];
  var y = [];
```

```

    for (var i in data){
        x.push(i);
        y.push(data[i]);
    }

    // line, bar, scatter diagram
    var plot_data = [{
        x: x, y: y, type: 'bar' // 'line','bar','scatter'
    }];
    var layout = {
        title: "Results in Each Log",
        xaxis:{
            title: "Log Name",
            showgrid: false,
            zeroline: false
        },
        yaxis:{
            title: "Appear Time",
            showgrid: false,
            zeroline: false
        }
    };

    // pie diagram
    // var plot_data = [{
    //     value: y, labels: x, type: 'pie'
    // }];
    // var layout = {height: 400, width: 500};

    Plotly.newPlot(div, plot_data, layout, {responsive:false});
}

```

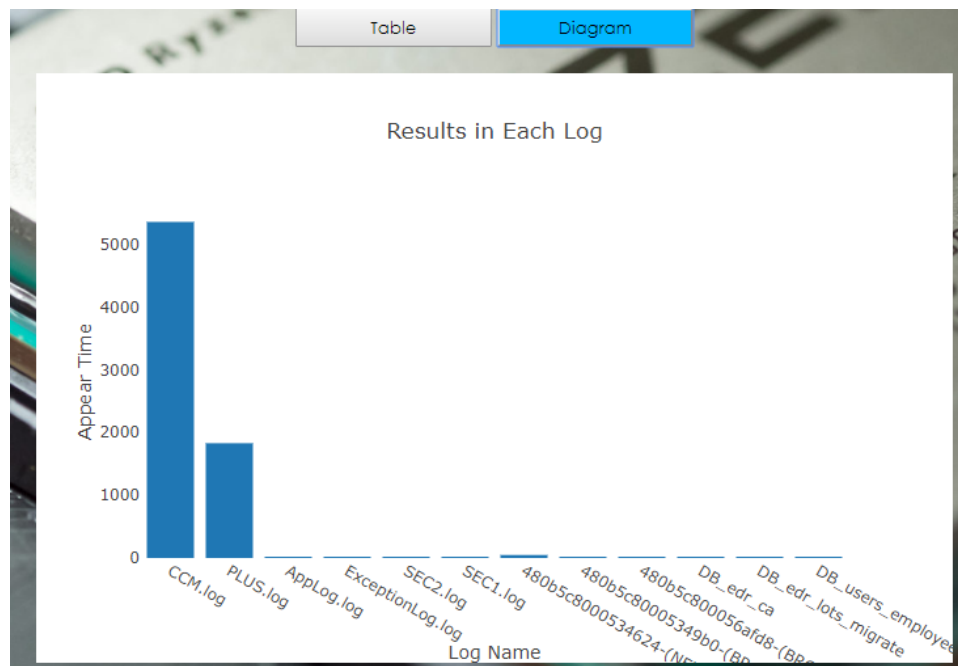


Diagram of result draw in Plotly.js

3. UI

Like the traditional web user-interface, JS + HTML + CSS is used in this system.

File:/stage3_v3/public/index_v3.html,

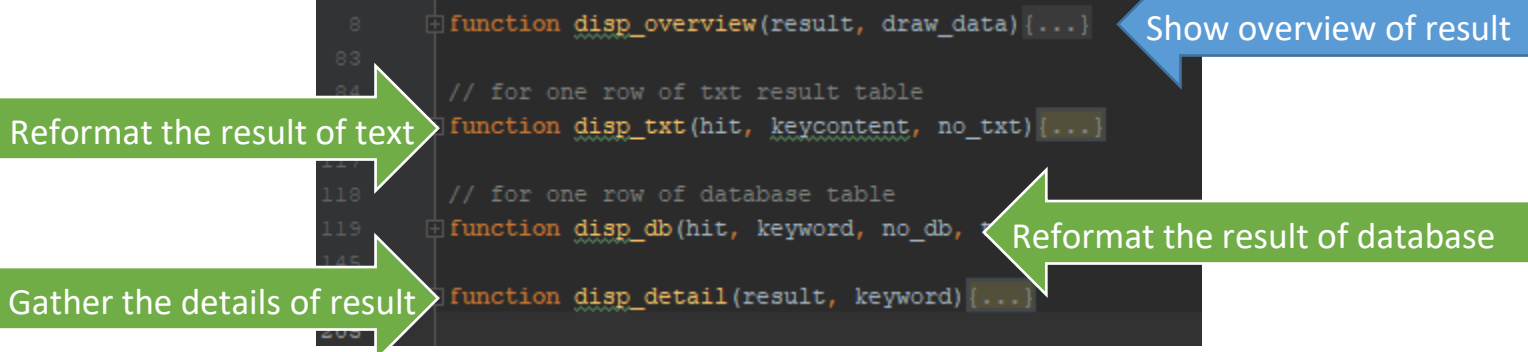
/stage3_v3/ui_v3.js

Note:

Use F12 in web browser to inspect the source code and the relative block.

a. ui.js

This part is for reformatting the search result and presenting on the HTML table in the webpage.



The structure of the ui.js

About 7,312 Results. Time Taken: 0.038 second.

Overview Result			
Log Folder	Log Name	Log Results Amount	Folder Results Amount
IT	CCM.log	5358	7244
	FLUS.log	1830	
	AppLog.log	16	
	ExceptionLog.log	16	
	SEC2.log	13	
	SEC1.log	11	
operator	480b5c8000534624-(NEW FGS SPLIT AUTO LABEL BACKEND VIRTUAL-BARTMP8)(LOTID LABEL AUTO VIRTUAL-VBARTMP2).txt	48	54
	480b5c80005349b0-(BRCM_PL OUTERBOX-BARTMP36).txt	3	
	480b5c800056cfd8-(BRCM_PL TNR-BARTMP36).txt	3	

Show the details of result

Detailed Result					
Number	Log Folder	Log Name	Log Date	Log Time	Message
0	IT	PLUS.log	11-Dec-2018	00:24:12.257	<pre> Received expected SECS II message with system byte 18038 573F74 W [[L [["2DSRT-02"]] [["PLUS_PMG3V_PROD"]]] [L [["HFLServer2_TMP"]] [["PLUS_PMG3V_PROD"]]] [L [U4 79] [U4 11] [U4 11] [L [L] [L ["header\x00Map"] [L ["JobID\x00String"] ["HFLServer2_TMP-2DSRT-02_2018-12-11_00:24:06.407_492"]]] [L ["RequestHdr\x00Map"] [L ["Additional\x00Content"]]]]]] </pre>
					<pre> Received expected SECS II message with system byte 18272 573F74 W [[L [["2DSRT-02"]] [["PLUS_PMG3V_PROD"]]] [L [["HFLServer2_TMP"]] [["PLUS_PMG3V_PROD"]]] [L [U4 79] [U4 11] [U4 11] [L [L] [L ["header\x00Map"] [L ["JobID\x00String"] ["HFLServer2_TMP-2DSRT-02_2018-12-11_00:24:06.407_492"]]] [L ["RequestHdr\x00Map"] [L ["Additional\x00Content"]]]]]] </pre>

b. index.html

The HTML part is for building blocks on webpage, nothing needs special illustration.

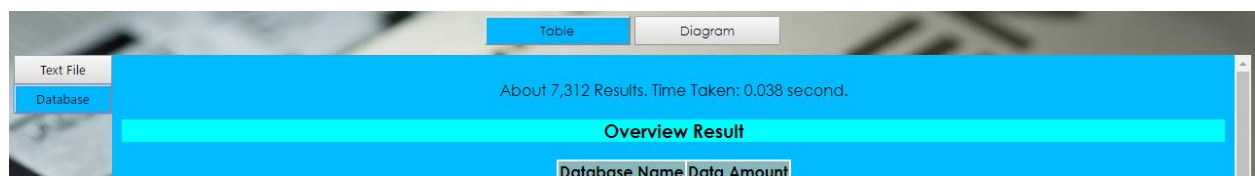
But there are some dynamic logic needs to be noticed.

The most important line of code is the form of

```
document.getElementById('input_keyword').style.display = 'block';
```

“”.style.display = ‘block’” means change the display status and make this element visible. If “”.style.display = ‘none’”, then this element is hidden.

Therefore, the dynamic logic part is mainly about change the display status so that change the subpage of presentation



Four buttons for four subpages

For instance, once the “butt_tab” element is clicked, the “nav_data_source” element is not hidden, and this button will change its own color to mark its status.

```

butt_tab.onclick = function () {
  // display area
  table.style.display = 'block';
  plot.style.display = 'none';
  document.getElementById('nav_data_source').style.display = 'block';
}

```



```
// change button status
butt_plot.className = '';
this.className = 'active';

document.getElementById('rt').style.backgroundColor = '#00BBFF';
};
```

c. CSS

Like the normal usage, the CSS is for setting style of the webpage elements.

There are several places need to be noticed:

```
button{
  font-family: "Yu Gothic UI";
  overflow: hidden;
}
button.active{ // change the button that class name is "active"
  background-color: #00B7FF;
}
button:hover{
  background-color: #00B7FF;
}
button:active { // set the response of mouse event
  background-color: #00B7FF;
  transform: translateY(2px);
}
```

Press F12 in web browser to inspect the element when you want to change the style of some elements.



The element id is "input"

Study Guide

For the sake of time, the future developer taking this project is suggested to flow this learning map drawn on the previous experience and errors.

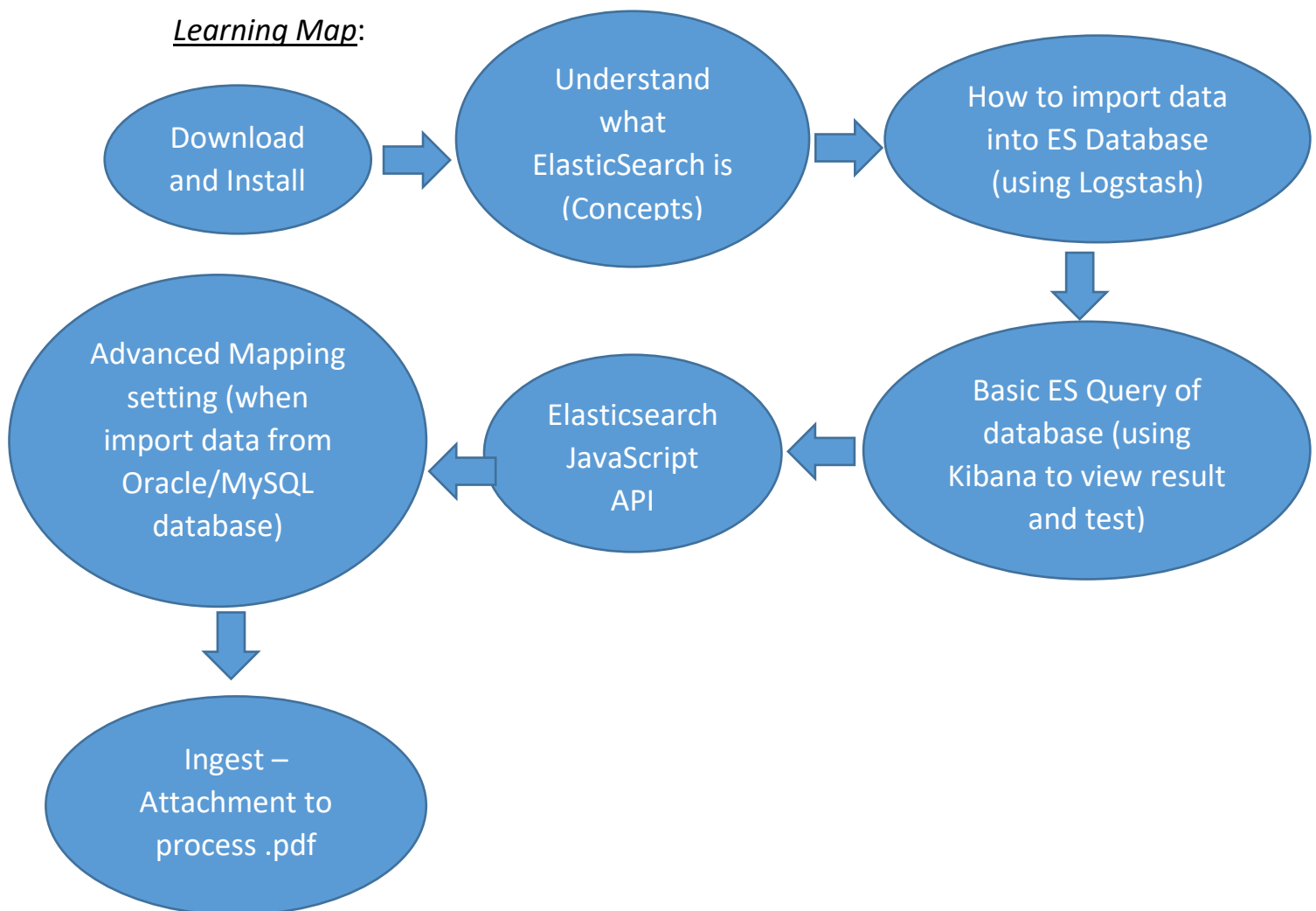
Back-End

The back-end of this system is mainly the search engine and data pre-processing system.

1. Elasticsearch

As the core of this system, the fundamental or even advanced knowledge is required at the first beginning in order to understand the basic requirement of this project.

Learning Map:



Reference:

Official Website:

<https://www.elastic.co/guide/cn/index.html>

<https://www.elastic.co/guide/index.html> (English)

Overall Introduce:

<https://blog.csdn.net/yezonggang/article/details/80064394>

Download and Installation:

<https://blog.csdn.net/weidong22/article/details/79062851>

Query (use combine with Kibana):

<https://www.elastic.co/guide/en/elasticsearch/client/javascript-api/current/quick-start.html> (English)

<https://blog.csdn.net/tototuzuoquan/article/details/78303095>

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-multi-match-query.html>

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-match-query-phrase-prefix.html> (English)

<http://n3xtchen.github.io/n3xtchen/elasticsearch/2017/07/05/elasticsearch-h-23-useful-query-example> (Recommend)

Blog Series (Recommend):

<http://www.cnblogs.com/ginb/p/6637236.html>

<http://www.cnblogs.com/ginb/p/elasticsearch.html>

<http://www.cnblogs.com/ginb/p/6993299.html>

<http://www.cnblogs.com/ginb/p/7000427.html>

JavaScript API:

<https://www.elastic.co/guide/en/elasticsearch/client/javascript-api/current/quick-start.html> (English)

Mapping:

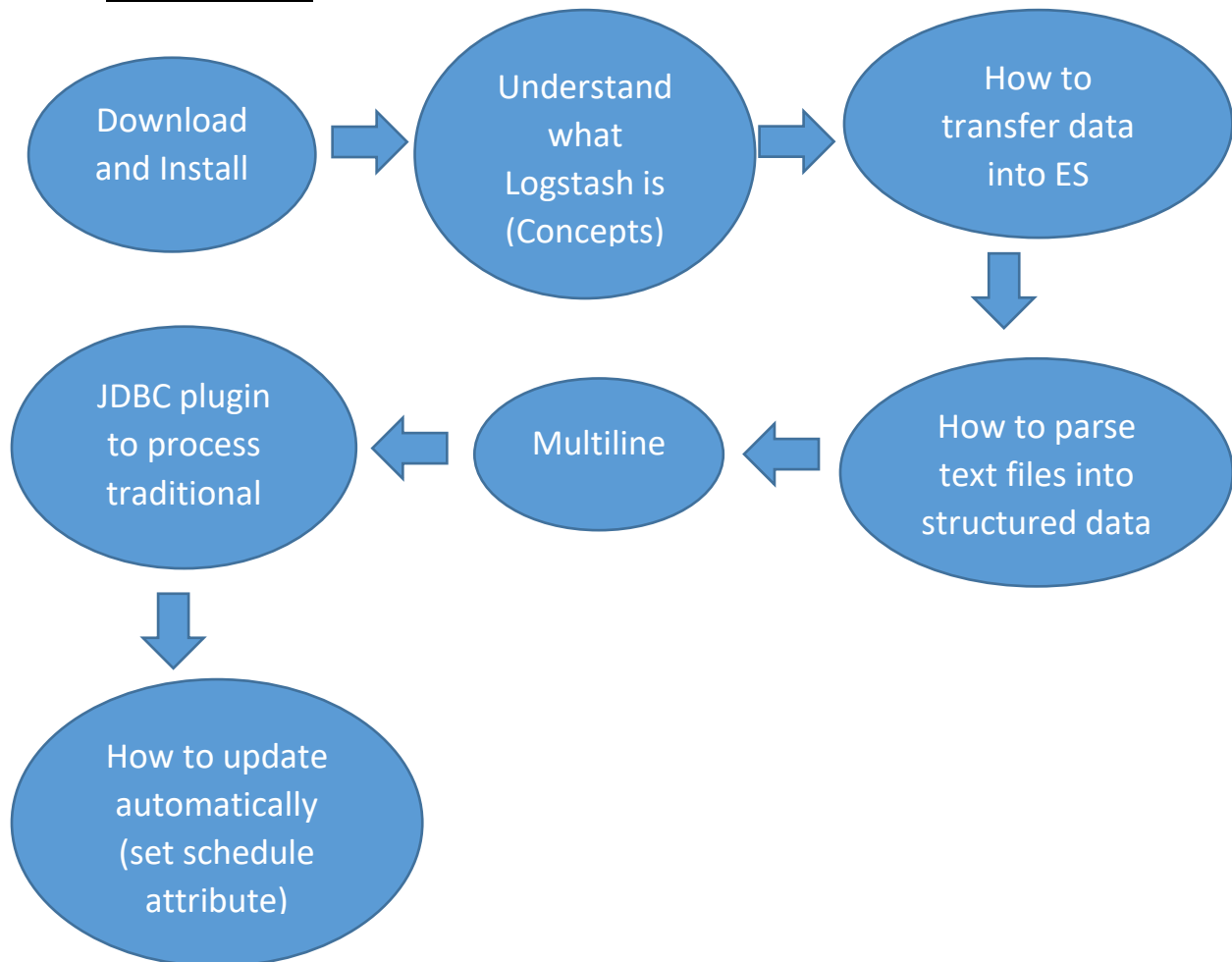
<https://my.oschina.net/davidzhang/blog/811511>

<https://stackoverflow.com/questions/37861279/how-to-index-a-pdf-file-in-elasticsearch-5-0-0-with-ingest-attachment-plugin?rq=1>

2. Logstash

Logstash is the tool to import and update data in Elasticsearch's database. It has the ability to parse unstructured files such as .log, .txt using regular expression. As well as transfer the data in existing Oracle/MySQL database into the Elasticsearch Database directly.

Learning Map:



Reference:

Official Website:

<https://www.elastic.co/products/logstash>

<https://www.elastic.co/guide/en/logstash/current/getting-started-with-logstash.html> (Series of Install, Guide) (English)

Basic Using:

<https://www.cnblogs.com/yincheng/p/logstash.html>

Parse Test Files:

<https://www.elastic.co/guide/en/logstash/6.5/advanced-pipeline.html>

(English)

<http://trumandu.github.io/2016/10/24/logstash%E4%BD%BF%E7%94%A8%E6%95%99%E7%A8%8B/>

Connect to ES:

<https://blog.csdn.net/wangnan9279/article/details/79287820>

Connect to Oracle Database (JDBC):

<https://blog.csdn.net/wjacketcn/article/details/50960843>

<https://blog.csdn.net/laoyang360/article/details/75452953>

<https://discuss.elastic.co/t/logstash-jdbc-input-oracle-settings/26996>

(English)

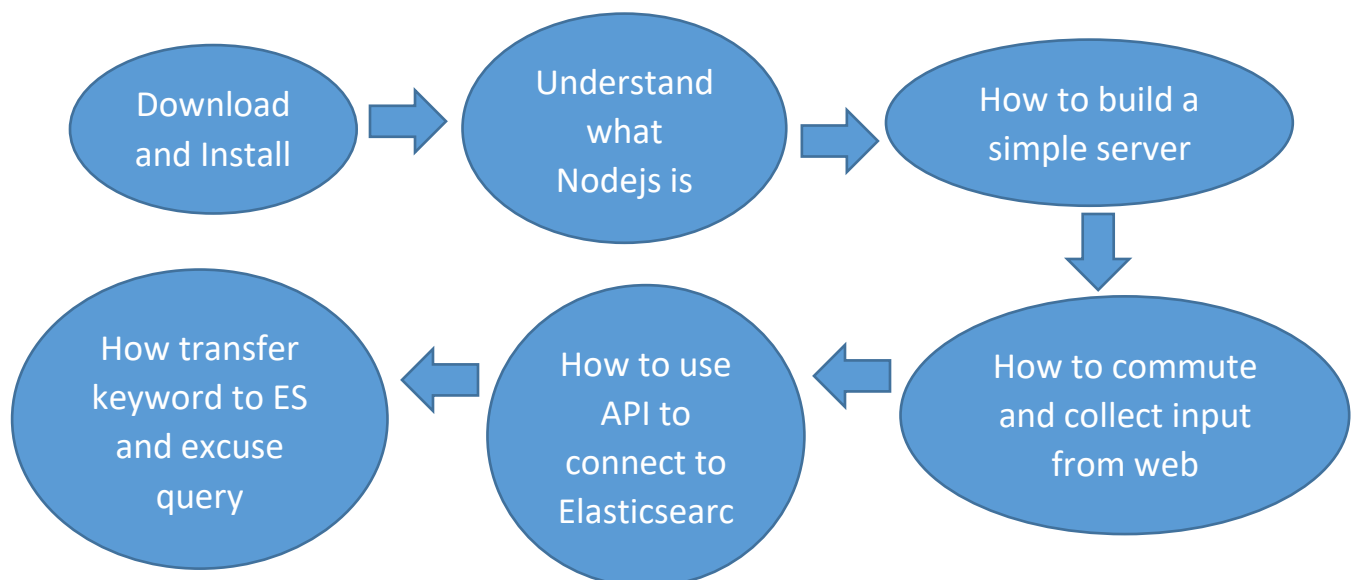
WebServer

The Webserver Part is responsible for transferring the keywords user inputs and return the result that the elasticsearch engine generates back to the front-end.

1. Nodejs

Nodejs is a powerful tool built on JavaScript, it's easy to operate and expand with other API such as Elasticsearch.

Learning Map:



Reference:

Official Website + Download:

<https://nodejs.org/en/> (English)

Introduce:

<https://codeburst.io/the-only-nodejs-introduction-youll-ever-need-d969a47ef219> (English)

Tutorial Series (Recommend):

<https://www.w3schools.com/nodejs/> (English)

Connect to ES:

<https://www.oschina.net/translate/search-engine-node-elasticsearch>

Asynchronization (Advanced):

<https://www.jb51.net/article/63070.htm>

<https://m.jb51.net/article/84148.htm>

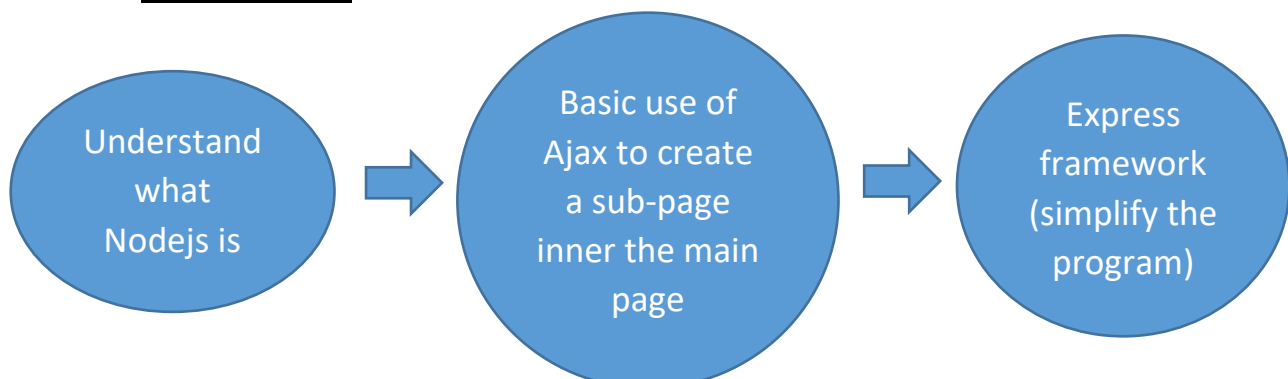
2. Ajax

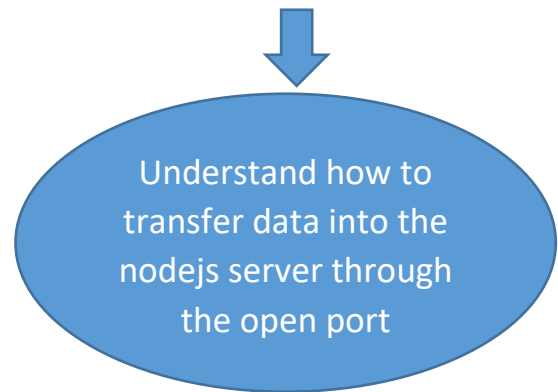
Ajax is a popular tool used when building webpage, here we also use it as the tool to transfer data between the web server and the front end.

```
// ajax data transfer
$('#show_all').click(function () {
  $.ajax({
    url: '/getkeyword',
    type: 'get',
    data: {
      show_all: "show_all"
    },
    success: function (data) {
      // parse the string data into Json
      var data_json = eval('(' + data + ')');
      // display result
    }
  });
});
```

Transfer data by Ajax combined with JQuery

Learning Map:





Reference:

Introduce:

https://www.w3schools.com/xml/ajax_intro.asp

Tutorial Series (Recommend):

https://www.w3schools.com/xml/ajax_intro.asp (English)

<http://www.runoob.com/ajax/ajax-tutorial.html>

Data Transmission:

<https://www.jb51.net/article/57874.htm> (Recommend)

3. JQuery

JQuery is a simple alternative of JavaScript, but it has lots of special functions and unique language format. It is used in this system to combine with Ajax.

It's not as important as other modules in this system, thus no detailed study of this section in this doc.

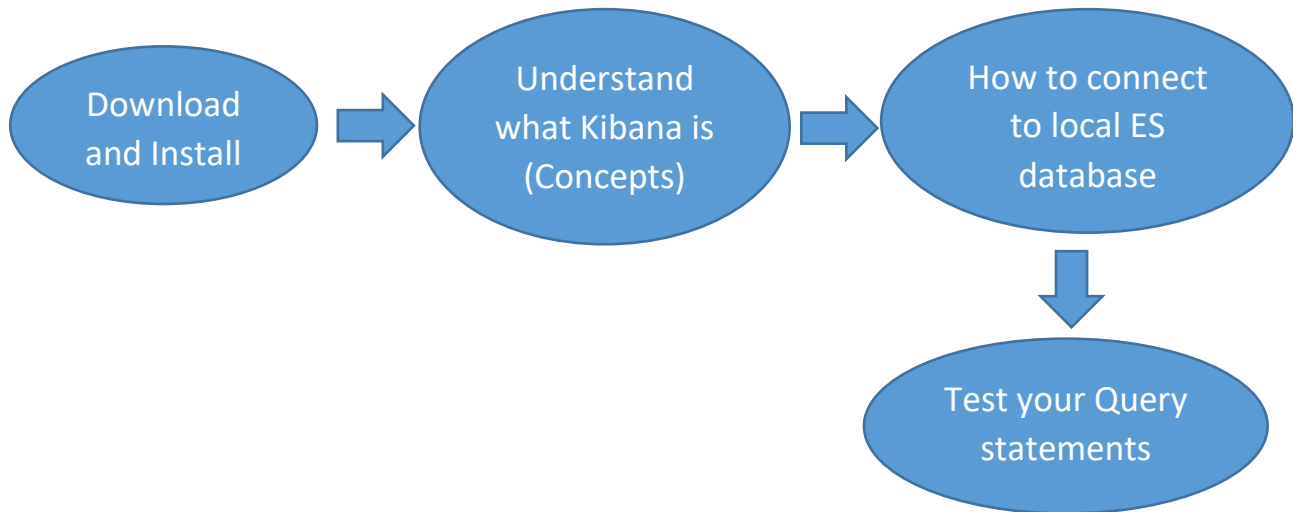
Front-End

The work of front-end is to reformat and present the search result that transferred by the server and generated by the back-end engine, as well as collect the keywords inputted by user.

1. Kibana

Kibana is part of the ELK (Elasticsearch – Logstash – Kibana) application stack. It is a tool to visualize the result, but it is replaced by our own webpage and only serve as the query – testing tool to inspect the elasticsearch database.

Learning Map:



Reference:

Official Website:

<https://www.elastic.co/guide/cn/kibana/current/index.html>

<https://www.elastic.co/guide/en/kibana/current/index.html> (English)

Download:

<https://www.elastic.co/downloads/kibana>

Connect with ES:

<https://www.elastic.co/guide/cn/kibana/current/connect-to-elasticsearch.html>

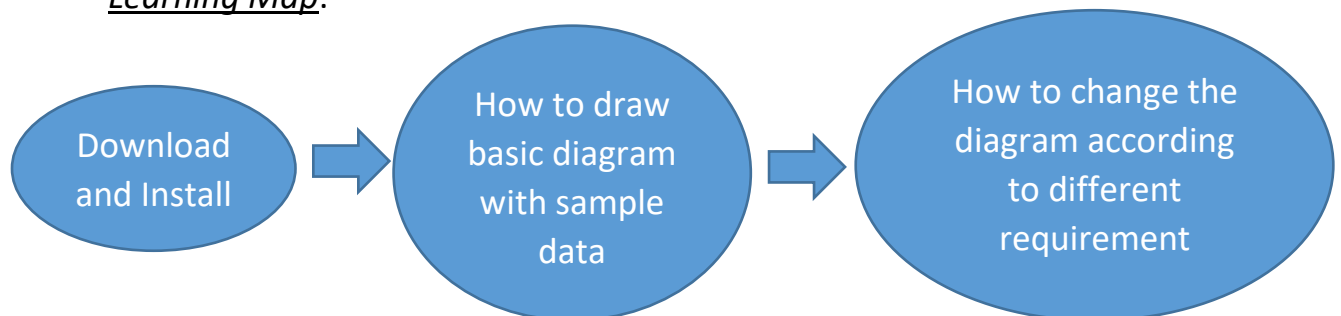
Tutorial:

<https://www.elastic.co/guide/en/kibana/current/index.html> (English)

2. Plotly

Plotly is a powerful and flexible graphic plugin built for JavaScript. It is used for visualization in this system.

Learning Map:



Reference:

Official Website, Download, Tutorial:

<https://plot.ly/javascript/>

Program Language


I believe a qualified software engineer is capable of handling any unfamiliar program language in very short term in grammar-level. Language is too easy to worry that much for a brilliant programmer. Therefore, this documentation dose not contain detailed guide of any program language, the reader could find various tutorials of basic language grammar knowledge in Google.

Configuration









This section is about how to active the whole system and some daily routine.

File Structure

The directory structure of the program is shown below.

 nodejs_project_AMD	1/15/2019 9:39 AM	File folder
--	-------------------	-------------

Root directory

hub > Elasticsearch-nodejs-UI > nodejs_project_AMD				Search nodejs_project_AMD	
Name	Date modified	Type	Size		
 conf	1/4/2019 3:43 PM	File folder			
 deliverable_s3_v3	1/8/2019 10:42 AM	File folder			
 nodes	1/4/2019 5:34 PM	File folder			
 sql	1/9/2019 2:11 PM	File folder			
 stage1_v3	1/2/2019 5:06 PM	File folder			
 stage2_v4	1/2/2019 5:06 PM	File folder			
 stage3_v3	1/14/2019 5:29 PM	File folder			
 handover.docx	1/15/2019 9:45 AM	Microsoft Word D...	394 KB		

Secondary directory

elasticsearch-nodejs-UI > nodejs_project_AMD > conf		Search conf
Name	Date modified	Type
logstash	1/4/2019 3:43 PM	File folder
mapping	1/4/2019 4:32 PM	File folder

/conf:

Contains the logstash configuration files needed when update data and mapping setting using in Kibana to set property of data fields (columns)

nodejs_project_AMD > deliverable_s3_v3		Search deliverable_s3_v3
Name	Date modified	Type
.idea	1/8/2019 10:49 AM	File folder
output	1/7/2019 2:01 PM	File folder
public	1/8/2019 10:42 AM	File folder
server_router_v3.js	1/14/2019 9:42 AM	JS File
server_searchengine_v3.js	1/14/2019 9:42 AM	JS File
ui_v3.js	1/14/2019 9:42 AM	JS File

Search result copy

Main server

HTML, CSS, images

/deliverable:

The latest version of the system, could be treated as the useable version for daily use, but development should not in here

hub > Elasticsearch-nodejs-UI > nodejs_project_AMD		Search nodejs_project_AMD
Name	Date modified	Type
conf	1/4/2019 3:43 PM	File folder
deliverable_s3_v3	1/8/2019 10:42 AM	File folder
nodes	1/4/2019 5:34 PM	File folder

/nodes:

Elasticsearch database copy

hub > Elasticsearch-nodejs-UI > nodejs_project_AMD				Search nodejs_project_AMD
Name	Date modified	Type	Size	
conf	1/4/2019 3:43 PM	File folder		
deliverable_s3_v3	1/8/2019 10:42 AM	File folder		
nodes	1/4/2019 5:34 PM	File folder		
sql	1/9/2019 2:11 PM	File folder		
stage1_v3	1/2/2019 5:06 PM	File folder		

/sql:

SQL statement, used when transfer data from Oracle/MySQL

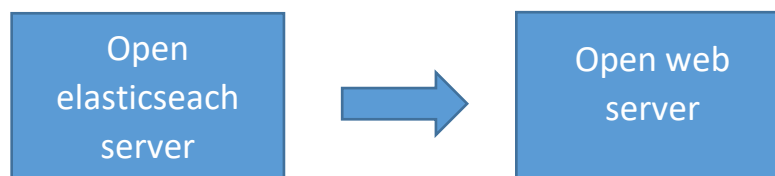
hub > Elasticsearch-nodejs-UI > nodejs_project_AMD				Search nodejs_project_AMD
Name	Date modified	Type	Size	
conf	1/4/2019 3:43 PM	File folder		
deliverable_s3_v3	1/8/2019 10:42 AM	File folder		
nodes	1/4/2019 5:34 PM	File folder		
sql	1/9/2019 2:11 PM	File folder		
stage1_v3	1/2/2019 5:06 PM	File folder		
stage2_v4	1/2/2019 5:06 PM	File folder		
stage3_v3	1/14/2019 5:29 PM	File folder		

/stage* v*:

History versions, development and system update should perform here

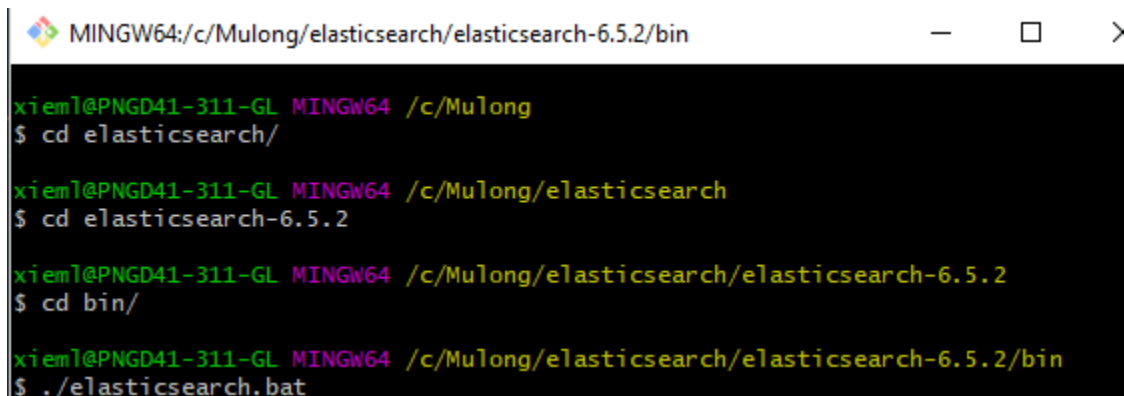
Deploy

When all current development is done, life becomes easy. To run the system, there are only two servers need to be active.



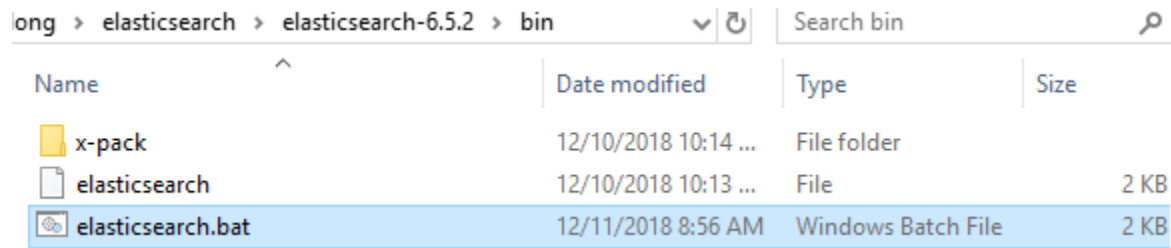
Open ES Server

After importing data, excuse the '.bat' to active the server before doing anything.



```
MINGW64:/c/Mulong/elasticsearch/elasticsearch-6.5.2/bin
xieml@PNGD41-311-GL MINGW64 /c/Mulong
$ cd elasticsearch/
xieml@PNGD41-311-GL MINGW64 /c/Mulong/elasticsearch
$ cd elasticsearch-6.5.2
xieml@PNGD41-311-GL MINGW64 /c/Mulong/elasticsearch/elasticsearch-6.5.2
$ cd bin/
xieml@PNGD41-311-GL MINGW64 /c/Mulong/elasticsearch/elasticsearch-6.5.2/bin
$ ./elasticsearch.bat
```

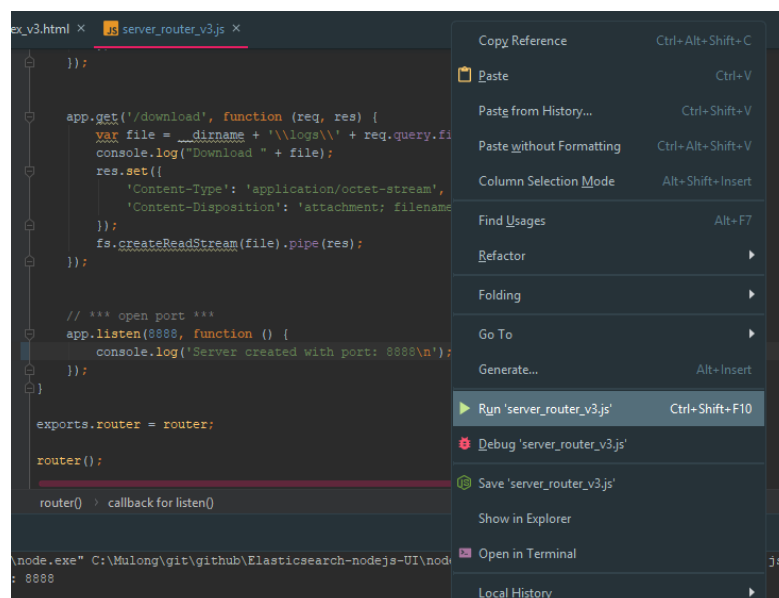
Or go to elasticsearch/bin and click the “.bat” file.



ong > elasticsearch > elasticsearch-6.5.2 > bin				Search bin	
Name	Date modified	Type	Size		
x-pack	12/10/2018 10:14 ...	File folder			
elasticsearch	12/10/2018 10:13 ...	File	2 KB		
elasticsearch.bat	12/11/2018 8:56 AM	Windows Batch File	2 KB		

Open Web Server

Open the main server “server_router”, and run it as running a normal node.js program.

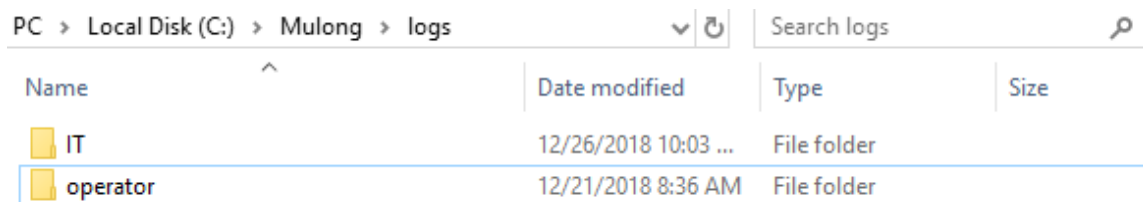


Data Update

Use Logstash to update the ES database.

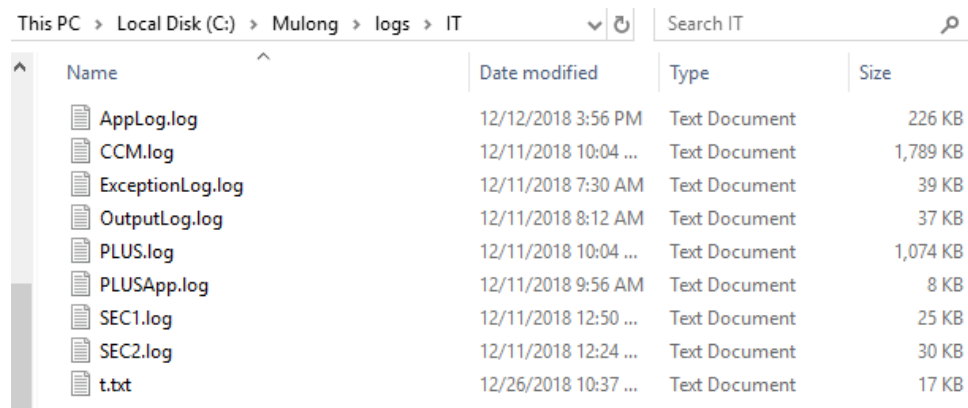
Update Text Files

1. Put new text file under any directory



PC > Local Disk (C:) > Mulong > logs				Search logs
Name	Date modified	Type	Size	
IT	12/26/2018 10:03 ...	File folder		
operator	12/21/2018 8:36 AM	File folder		

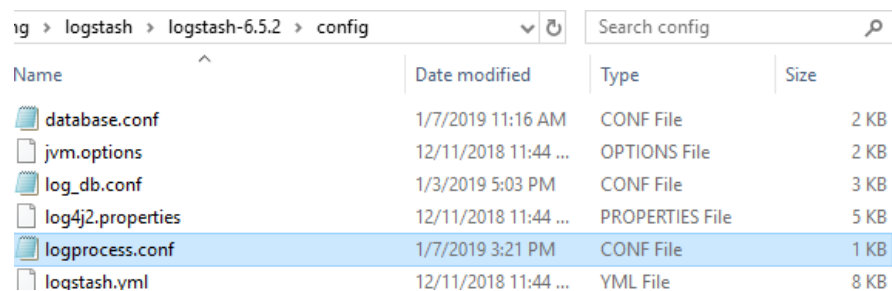
The current folders



This PC > Local Disk (C:) > Mulong > logs > IT				Search IT
Name	Date modified	Type	Size	
AppLog.log	12/12/2018 3:56 PM	Text Document	226 KB	
CCM.log	12/11/2018 10:04 ...	Text Document	1,789 KB	
ExceptionLog.log	12/11/2018 7:30 AM	Text Document	39 KB	
OutputLog.log	12/11/2018 8:12 AM	Text Document	37 KB	
PLUS.log	12/11/2018 10:04 ...	Text Document	1,074 KB	
PLUSApp.log	12/11/2018 9:56 AM	Text Document	8 KB	
SEC1.log	12/11/2018 12:50 ...	Text Document	25 KB	
SEC2.log	12/11/2018 12:24 ...	Text Document	30 KB	
t.txt	12/26/2018 10:37 ...	Text Document	17 KB	

Put new file here

2. Open [logstash/config/logprocess.conf](#)



log > logstash > logstash-6.5.2 > config				Search config
Name	Date modified	Type	Size	
database.conf	1/7/2019 11:16 AM	CONF File	2 KB	
jvm.options	12/11/2018 11:44 ...	OPTIONS File	2 KB	
log_db.conf	1/3/2019 5:03 PM	CONF File	3 KB	
log4j2.properties	12/11/2018 11:44 ...	PROPERTIES File	5 KB	
logprocess.conf	1/7/2019 3:21 PM	CONF File	1 KB	
logstash.yml	12/11/2018 11:44 ...	YML File	8 KB	

3. Change few lines of configuration

And save it under /sql

sticsearch-nodejs-UI > nodejs_project_AMD > sql				
Search sql				
Name	Date modified	Type	Size	
1.sql	12/28/2018 5:17 PM	Microsoft SQL Ser...	1 KB	
2.sql	12/28/2018 5:16 PM	Microsoft SQL Ser...	1 KB	
3.sql	12/28/2018 5:18 PM	Microsoft SQL Ser...	1 KB	
4.sql	1/3/2019 11:44 AM	Microsoft SQL Ser...	1 KB	
5.sql	1/4/2019 4:00 PM	Microsoft SQL Ser...	1 KB	
6.sql	1/4/2019 4:07 PM	Microsoft SQL Ser...	1 KB	
7.sql	1/4/2019 4:14 PM	Microsoft SQL Ser...	1 KB	
8_mass.sql	1/7/2019 9:06 AM	Microsoft SQL Ser...	1 KB	
9_mass.sql	1/7/2019 11:00 AM	Microsoft SQL Ser...	1 KB	
10_mass.sql	1/7/2019 11:10 AM	Microsoft SQL Ser...	1 KB	
11_mass.sql	1/7/2019 11:10 AM	Microsoft SQL Ser...	1 KB	
test.sql	1/9/2019 2:12 PM	Microsoft SQL Ser...	1 KB	

2. Change the config file.

ong > logstash > logstash-6.5.2 > config				
Search config				
Name	Date modified	Type	Size	
database.conf	1/7/2019 11:16 AM	CONF File	2 KB	

database.conf - Notepad

```
File Edit Format View Help
input{
  jdbc{
    jdbc_driver_library => "C:\Mulong\jdbc\ojdbc8.jar"
    jdbc_driver_class => "Java:oracle.jdbc.driver.OracleDriver"
    jdbc_connection_string => "jdbc:oracle:thin:edr_admin/edr_admin32229@//vpngorasvdlstg:1521/svdlqa"
    jdbc_user => "edr_admin"
    jdbc_password => "edr_admin32229"
    statement_filepath => "C:\Mulong\git\github\Elasticsearch-nodejs-UI\nodejs_project_AMD\sql\9_mass.sql"
    type => "edr_audit"
  }
  jdbc{
    jdbc_driver_library => "c:\Mulong\jdbc\ojdbc8.jar"
    jdbc_driver_class => "Java:oracle.jdbc.driver.OracleDriver"
    jdbc_connection_string => "jdbc:oracle:thin:edr_admin/edr_admin32229@//vpngorasvdlstg:1521/svdlqa"
    jdbc_user => "edr_admin"
    jdbc_password => "edr_admin32229"
    statement_filepath => "C:\Mulong\git\github\Elasticsearch-nodejs-UI\nodejs_project_AMD\sql\10_mass.sql"
    type => "edr_lots_audit"
  }
  jdbc{
    jdbc_driver_library => "c:\Mulong\jdbc\ojdbc8.jar"
    jdbc_driver_class => "Java:oracle.jdbc.driver.OracleDriver"
    jdbc_connection_string => "jdbc:oracle:thin:edr_admin/edr_admin32229@//vpngorasvdlstg:1521/svdlqa"
    jdbc_user => "edr_admin"
    jdbc_password => "edr_admin32229"
    statement_filepath => "C:\Mulong\git\github\Elasticsearch-nodejs-UI\nodejs_project_AMD\sql\11_mass.sql"
    type => "edr_lots_bkp"
  }
}
```

Copy this jdbc for every new table

New table name

New SQL

3. Excuse the new configuration.

```
C:\Mulong\logstash\logstash-6.5.2\bin>logstash -f ../config/dbprocess.conf
```

Future Challenge

For the reason of time, this system is still not perfect and has zone to improve and expand.

The challenges here are all advanced topics, the future solver is required insight and comprehension of previous knowledge before going ahead.

PDF Functionality

A new requirement is searching in .pdf file, but based on current knowledge, it is inevitable to convert the pdf files one by one before importing into ES database.

The future developer is suggested to use some other languages to write a script to achieve conversion and then pipeline to ES.

More Diagrams

The visualization functionality is now a simple demo, the future taker might can draw various diagrams to meet different needs.

The future developer is suggested to not only learn how to draw different diagram, also need to know how to expand the user interface to present those graphs in a user-friendly way.

Search Accuracy

The system is still now using testing data sample, but for the sake of time, all the fields of database is reset as “text”, which may bring some unwanted results.

The future developer is suggested to learn to master “match_phrase”, “term”, “query_string” and other advanced query methods.

It is very import to consider carefully what type or multiple-types should be assigned to different data filed when setting a new mapping.

Database Presentation

A non-important drawback of this system is that only limited result could be presented on webpage.

The future developer is suggested to add “show more” functionality.