Studying MongoDB, Elasticsearch and NPM

RAJESH NATH | CHARU PUJARA | BINDIYA AHUJA

A joint paper by JP Tokyo & Co and MRIIRS

Ms. Tanvi Sharma

2018

Contents

[Introduction 2](#_Toc536827296)

[Pre-Requisites 2](#_Toc536827297)

[Platforms Used 3](#_Toc536827298)

[Operating System 3](#_Toc536827299)

[Node Package Manager 3](#_Toc536827300)

[MongoDB 3](#_Toc536827301)

[Elastic Search 3](#_Toc536827302)

[Connecting to MongoDb 4](#_Toc536827303)

[Using the Mongo Connector 4](#_Toc536827304)

[Elastic Search Setup 4](#_Toc536827305)

[Front-end 5](#_Toc536827306)

[References 7](#_Toc536827307)

# Introduction

This paper explains about the use of technologies like NodeJS, Elasticsearch algorithms and MongoDB for creating a project for fast, autofill searches on our Database. NodeJS accesses project related data stored on MongoDB, using Elastic Search technique which makes the searches easier and faster.

The project is about creating databases, manipulating them. It includes creating applications to use the databases and connecting the servers.

The data directories are saved on MongoDB, searches are done using Elastic Search and frontend programming is done in NodeJS. The platform OS used is Ubuntu.

## Pre-Requisites

The project the following environments to start

* A working installation of MongoDB platform.
* Start the server as a replica set
* Use the command, “mongod --replSet rs0”
* In case of error, use [rs.intiate()](https://docs.mongodb.com/manual/reference/method/rs.initiate/) (ctrl+click on the command for info)
* Working Elastic Search Binaries
* Python with pip installed

Why Replica?

It makes a duplicate of the original. These replicas are the slaves in which the changes done are untouched by the master even while syncing.

# Platforms Used

## Operating System

This project uses Linux based OS, Ubuntu mate version 18.04. Ubuntu is preferred for it is Open Source, and has a strong and helpful community. The community much helps in providing solutions for problems faced in making project. It is easy to use for it has everything required already installed and just a command away on the terminal. Ubuntu can be used from desktop to cloud and all connected devices.

## Node Package Manager

NPM is the package manager for JavaScript language. It is the default package manager for the JavaScript runtime environment [Node.js](https://en.wikipedia.org/wiki/Node.js). It consists of a command line client and an [online database](https://en.wikipedia.org/wiki/Online_database) of public and paid-for private packages. The registry is accessed via the client, and the available packages can be browsed and accessed via the npm website. It is written completely in JavaScript and has the largest Software Registry.

## MongoDB

MongoDB is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [document-oriented database](https://en.wikipedia.org/wiki/Document-oriented_database) program. MongoDB uses [JSON](https://en.wikipedia.org/wiki/JSON)-like documents with [schemata](https://en.wikipedia.org/wiki/Database_schema) and classified as a [NoSQL](https://en.wikipedia.org/wiki/NoSQL) database program, It is a nonrelational [database](https://searchsqlserver.techtarget.com/definition/database) technology as the MongoDB architecture is made up of collections and documents, rather than rows and columns. MongoDb stores data in flexible, JSON-like documents.

## Elastic Search

Elasticsearch is search engine made on Lucene Language. It gives very distributed, multitenant-capable full-text search engine with Hyper Text Transfer Protocol web interface and schema-free JSON documents. Elasticsearch is developed in Java. It is an Open Source Software, licensed under Apache.

# Connecting to MongoDb

The replica set were connected by using Mongo connector to Elasticsearch. Mongo connector is a python-based library which keeps insync with indexes of MongoDb and Elasticsearch.

Indexes - Elasticsearch helps in searching and for that we need indexes. To keep it InSync with MongoDb we kept replica set.

The Following code has been used to achieve the connection to MongoDB in Linux,

# Start MongoDB session

mongo

# Create a database named jp\_tokyo

use jp\_tokyo

# Create a collection named products

db.createCollection('products')

## Using the Mongo Connector

The following code represents how to connect

mongo-connector -m 127.0.0.1:27017 -t 127.0.0.1:9200 -d elastic2\_doc\_manager -n jp\_tokyo.products -g products\_opt.products

# Elastic Search Setup

The use of auto complete requires an analyser chain. This is accomplished by using one of the many analysers provided by Elasticsearch. This project the [edge\_ngrams](https://en.wikipedia.org/wiki/N-gram) analyser is used due to its ease of understanding and easy set-up.

# Front-end

<!doctype html><html lang="en">

<head>

<meta charset="utf-8">

<link rel="shortcut icon" href="/favicon.ico">

<meta name="viewport" content="width=device-width,initial-scale=1,shrink-to-fit=no"><meta name="theme-color" content="#000000"><link rel="manifest" href="/manifest.json">

<title>React App</title>

<link śhref="/static/css/1.cd0e9c04.chunk.css" rel="stylesheet">

</head>

<body>

<noscript>You need to enable JavaScript to run this app.</noscript>

<div id="root"></div>

<script>!function(l){function e(e){for(var r,t,n=e[0],o=e[1],u=e[2],f=0,i=[];

f<n.length;f++)t=n[f],p[t]&&i.push(p[t][0]),p[t]=0;

for(r in o)

Object.prototype.hasOwnProperty.call(o,r)&&(l[r]=o[r]);

for(s&&s(e);i.length;)i.shift()();

return c.push.apply(c,u||[]),a()}

function a(){for(var e,r=0;r<c.length;r++){for(var t=c[r],n=!0,o=1;o<t.length;o++){var u=t[o];0!==p[u]&&(n=!1)}n&&(c.splice(r--,1),e=f(f.s=t[0]))}return e}var t={},p={2:0},c=[];

function f(e){if(t[e])return t[e].exports;

var r=t[e]={i:e,l:!1,exports:{}};

return l[e].call(r.exports,r,r.exports,f),r.l=!0,r.exports}

f.m=l, f.c=t, f.d=function(e,r,t){f.o(e,r)||Object.defineProperty(e,r,{enumerable:!0,get:t})},

f.r=function(e){"undefined"!=typeof Symbol&&Symbol.toStringTag&&Object.defineProperty(e,Symbol.toStringTag, {value:"Module"}),Object.defineProperty(e,"\_\_esModule",{value:!0})}, f.t=function(r,e){if(1&e&&(r=f(r)),8&e)

return r;

if(4&e&&"object"==typeof r&&r&&r.\_\_esModule)

return r;

var t=Object.create(null);

if(f.r(t),Object.defineProperty(t,"default", {enumerable:!0,value:r}),2&e&&"string"!=typeof r)

for(var n in r)f.d(t,n,function(e){return r[e]}.bind(null,n));

return t},

f.n=function(e)

{var r=e&&e.\_\_es

Module?function(){return e.default}: function(){return e};

return f.d(r,"a",r),r},

f.o=function(e,r)

{return Object.prototype.hasOwnProperty.call(e,r)},f.p="/";

var r=window.webpackJsonp||[],n=r.push.bind(r);

r.push=e,r=r.slice();

for(var o=0;o<r.length;o++)e(r[o]);var s=n;

a()}([])</script>

<script src="/static/js/1.3efe50af.chunk.js"></script>

<script src="/static/js/main.ea5d578a.chunk.js"></script>

</body>

</html>

# References

<https://www.youtube.com/watch?v=kwxGbO6Yzvc&feature=youtu.be>