1. NPM初始化
   1. Mkdir findajob
   2. Cd findajob
   3. Npm init// create a package.json
2. Npm install express body-parser elasticsearch
3. Index.js template.html

|  |
| --- |
| //index.js  // 需要Elasticsearch librray  **const** elasticsearch = require('elasticsearch');  // 实例化一个elasticsearch客户端  **const** client = **new** elasticsearch.Client({  hosts: [ 'http://localhost:9200']  });  //require Express  **const** express = require( 'express' );  // 实例化一个表达式的实例并将其保存在一个名为app的常量中  **const** app = express();  // 引入body-parser库。将用于解析主体请求  **const** bodyParser = require('body-parser')  //require the path library  **const** path = require( 'path' );  // ping客户端以确保Elasticsearch已启动  client.ping({  requestTimeout: 30000,  }, **function**(error) {  // 此时，eastic搜索已关闭，请检查您的Elasticsearch服务  **if** (error) {  console.error('elasticsearch cluster is down!');  } **else** {  console.log('Everything is ok');  }  });  // 使用bodyparser作为中间件  app.use(bodyParser.json())  // 设置应用程序侦听的端口  app.set( 'port', process.env.PORT || 3001 );  // 设置路径来提供静态文件  app.use( express.static( path.join( \_\_dirname, 'public' )));  // 启用CORS  app.use(**function**(req, res, next) {  res.header("Access-Control-Allow-Origin", "\*");  res.header('Access-Control-Allow-Methods', 'PUT, GET, POST, DELETE, OPTIONS');  res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");  next();  });  // 定义了基本路线并返回一个名为tempate.html的HTML文件  app.get('/', **function**(req, res){  res.sendFile('template.html', {  root: path.join( \_\_dirname, 'views' )  });  })  // 定义应该返回弹性搜索结果的/ search路径  app.get('/search', **function** (req, res){  // 声明查询对象以搜索弹性搜索，并从找到的第一个结果中仅返回200个结果。  // 还匹配其中名称与发送的查询字符串类似的任何数据  **let** body = {  size: 200,  from: 0,  query: {  match: {  name: req.query['q']  }  }  }  // 在索引中执行实际的搜索传递，搜索查询和类型  client.search({index:'cn\_jobs', body:body, type:'cn\_jobs'})  .then(results => {  res.send(results.hits.hits);  })  .catch(err=>{  console.log(err)  res.send([]);  });  })  // 监听一个指定的端口  app .listen( app.get( 'port' ), **function**(){  console.log( 'Express server listening on port ' + app.get( 'port' ));  } ); |
| <link rel="stylesheet" type="text/css" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">  <script src="https://unpkg.com/axios/dist/axios.min.js"></script>  <script src="https://cdn.jsdelivr.net/npm/vue"></script>  <div>  <div>  <div>  <h1>Search Cities around the world</h1>  </div>  </div>  <div>  <div>  <form action="" class="search-form">  <div>  <label for="search" class="sr-only">Search</label>  <input type="text" class="form-control" name="search" id="search" placeholder="search" v-model="query" >  <span></span>  </div>  </form>  </div>  </div>  <div>  <div>  <div>  <div>    {{ result.\_source.name }}, {{ result.\_source.country }}  </div>  <div>    <p>lat:{{ result.\_source.lat }}, long: {{ result.\_source.lng }}.</p>  </div>  </div>  </div>  </div>  </div>  <style>  .search-form .form-group {  float: right **!important**;  transition: all 0.35s, border-radius 0s;  width: 32px;  height: 32px;  background-color: #fff;  box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset;  border-radius: 25px;  border: 1px solid #ccc;  }  .search-form .form-group **input**.form-control {  padding-right: 20px;  border: 0 none;  background: transparent;  box-shadow: none;  display: block;  }  .search-form .form-group **input**.form-control::-webkit-input-placeholder {  display: none;  }  .search-form .form-group **input**.form-control:-moz-placeholder {  /\* Firefox 18- \*/  display: none;  }  .search-form .form-group **input**.form-control::-moz-placeholder {  /\* Firefox 19+ \*/  display: none;  }  .search-form .form-group **input**.form-control:-ms-input-placeholder {  display: none;  }  .search-form .form-group:hover,  .search-form .form-group.hover {  width: 100%;  border-radius: 4px 25px 25px 4px;  }  .search-form .form-group **span**.form-control-feedback {  position: absolute;  top: -1px;  right: -2px;  z-index: 2;  display: block;  width: 34px;  height: 34px;  line-height: 34px;  text-align: center;  color: #3596e0;  left: initial;  font-size: 14px;  }  </style>  // template.html  // 创建一个新的Vue实例  **var** app = **new** Vue({  el: '#app',  // 声明组件的数据（容纳结果的数组以及包含当前搜索字符串的查询） search string)  data: {  results: [],  query: ''  },  // 在这个Vue组件中声明方法。这里只定义了一种执行搜索的方法  methods: {  // 使用当前搜索查询向服务器发出axios请求  search: **function**() {  axios.get("http://127.0.0.1:3001/search?q=" + **this**.query)  .then(response => {  **this**.results = response.data;  })  }  },  // declare Vue watchers  watch: {  // 注意查询字符串中的更改并调用搜索方法  query: **function**() {  **this**.search();  }  }  }) |
|  |
|  |
|  |
|  |
|  |
|  |