# 动力云客项目

# 项目概述

动力云客项目是一款商业的集营销销售为一体的客户关系管理系统，其采用信息化、数字化方式来进行营销销售及客户管理；

动力云客，动力即指动力节点，云客即指海量客户，通过技术方式实现的这一套系统，可用于自动化分析销售、市场营销、客户服务等各个流程，建立起以客户为中心的信息化管理，从而支持更加有效的市场营销、销售与服务等各个环节，提高效率，提高效益；



# 项目技术选型及开发工具

前后端分离的项目（前端项目 + 后端项目）

* 前端：Html、CSS、JavaScript、Vue、Axios、Element Plus
* 后端：Spring Boot、Spring Security、MyBatis、MySQL、Redis
* 相关组件：HiKariCP（Spring Boot默认数据库连接池）、Spring-Data-Redis（Spring整合Redis）、Lombok（代码生成工具）、jwt（Json web token）、EasyExcel（Excel处理类库）、ECharts（前端图表库）
* 服务器：MySQL、Redis、Linux
* 项目依赖管理：Maven
* 项目开发工具：IDEA、Apifox

注意：项目开发一定要自己动手写，千万不能只看不写；

如果前期有困难，也一定要坚持写，突破了瓶颈后就好了；

# 项目数据库

在Linux MySQL数据库中创建一个名叫dlyk的数据库；

dlyk.sql （导入这个sql脚本）

# 前端项目

前后端分离项目 （两个项目，一个前端，一个后端）

前端项目（Vue） --->ajax请求(http请求) ---> 后端项目（Spring Boot）

## 前端环境工具准备

**node.js、 npm、 Vite （三个工具）**

**1、node.js**是一个开源、跨平台的 JavaScript 运行时环境，可以理解成java当中需要安装的jdk；

官网：<https://nodejs.org/en>

**2、npm**是JavaScript依赖包管理工具，全世界都可以用它来共享javascript包，负责前端项目的打包，插件下载等，可以理解成java中的maven；

官网：<https://www.npmjs.com/>

**3、Vite**是快速构建前端Vue项目的脚手架，可以理解为开发Spring Boot的Spring Initializr快速构建工具；

官网：<https://cn.vitejs.dev/>

|  |  |
| --- | --- |
| node.js | jdk |
| npm | maven |
| Vite | Spring Initializr |

## 安装Node.js

下载：<https://nodejs.org/en/download>

安装：解压即完成安装；node-v20.10.0-win-x64.zip版本；

## Nodejs环境变量配置

在 D:\course\tool\node-v20.10.0-win-x64 下创建2个文件夹：

node\_global （依赖库）

node\_cache （缓存）

然后在当前目录下cmd进入dos窗口，执行：

npm config set prefix "D:\course\tool\node-v20.10.0-win-x64\node\_global"

npm config set cache "D:\course\tool\node-v20.10.0-win-x64\node\_cache"

然后你可以查一下，看看有没有设置成功：

npm config get prefix

npm config get cache

在windows环境变量配置Path，在path变量中添加如下路径：

D:\course\tool\node-v20.10.0-win-x64

D:\course\tool\node-v20.10.0-win-x64\node\_global

在任意目录打开dos窗口测试下：

node -v

npm -v

## 配置npm仓库

|  |  |
| --- | --- |
| node.js | jdk |
| npm | maven |
| vite | Spring Initializr |

新版的Node.js已经自带npm，安装Node.js时就已经安装npm，所以不需要再安装npm，只需要配置一下npm仓库源即可；

查看目前的npm仓库源（位置）：

npm config get registry

在使用npm命令时，如果直接从国外的仓库https://www.npmjs.com/ 下载依赖，下载速度很慢，甚至会下载失败，我们可以更换npm的仓库源，提高下载速度，淘宝给我们提供了一个npm源；

//设置淘宝源

npm config set registry <https://registry.npmmirror.com/>

原来的老的淘宝npm地址（http://registry.npm.taobao.org/）即将停止解析，所以建议使用新地址，参考https://npmmirror.com/的页面说明；

安装一个模块（js依赖库）测试一下看看：

npm install axios -g #其中-g是全局安装的意思；

npm i axios -g #其中-g是全局安装的意思；

全局安装就会把axios模块安装到；

E:\course\tool\node-v20.9.0-win-x64\node\_global目录下，如果不加-g就会安装到了当前执行命令所在的目录下：C:\Users\Administrator

## Vite安装

vite是vue.js的脚手架，用于自动生成vue.js的项目模板（项目基础骨架）；

官网：<https://cn.vitejs.dev/> ( 之前使用Vue-Cli )

npm create vite@latest

npm是Node Package Manager的缩写，是Node.js的一个包管理工具。create是一个npm的命令，用于创建新的npm包。vite是一个基于Vue.js的静态网站生成器，@latest表示使用最新版本的vite，即npm create vite@latest为创建一个使用最新版本vite的npm包；

## Vue项目开发工具

项目代码开发工具

1、VScode （全称Visual Studio Code）

2、WebStorm；（也是idea公司出品的）

3、Idea (新版本的idea自带了vue插件)

4、Hbuilder；

# 部署原来的项目

1. 把老代码压缩包解压；
2. 把解压后得到的前后端两个项目，用idea打开；
3. idea修改以下文件编码为utf-8，设置一下maven；
4. 启动后端项目，启动前修改一下mysql、redis的连接信息；
5. 启动前端项目；（配置好idea里面的nodejs）

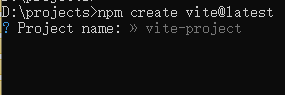
# 登录模块开发

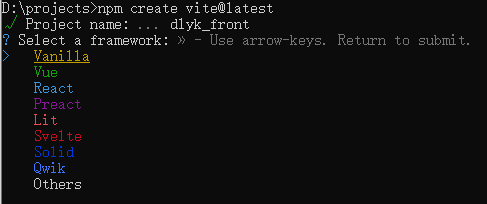
先写前端，再写后端；

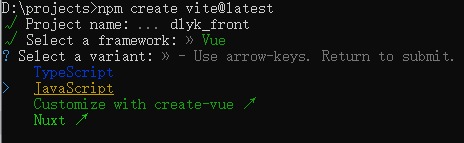
## 创建Vue项目工程

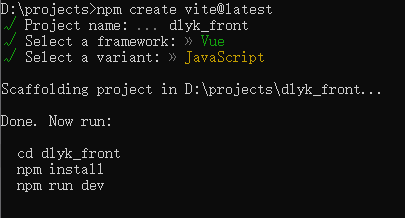
采用vite脚手架工具创建Vue项目工程；

npm create vite@latest

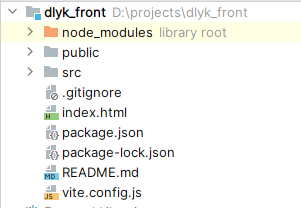








## Vue项目工程结构说明



1、node\_modules/ 项目依赖的各种js依赖包；

2、public/ 公共的静态文件，一个网站图标；

3、src/ 源代码，我们前端写的源代码都会在这个文件夹下；

4、gitignore 提交git时忽略哪些文件；

5、index.html 项目的首页，访问入口文件；

6、package.json 整个项目对js依赖库的配置，还包括了启动、构建项目命令等；（类似maven的pom.xml）

7、package-lock.json 锁定各个js依赖包的来源和版本；

8、README.md 项目的使用帮助文档；

9、vite.config.js vite的配置文件；

## 启动Vue项目工程

原来老的vue前端项目占用了8080，启动前修改一下启动端口：

|  |
| --- |
| import { *defineConfig* } from 'vite' import *vue* from '@vitejs/plugin-vue' export default *defineConfig*({  plugins: [*vue*()],   server:{  host:'0.0.0.0' ,*//ip地址* port: 8081, *// 设置服务启动端口号* open: true, *// 设置服务启动时是否自动打开浏览器* } }) |

命令行启动：

|  |
| --- |
| cd dlyk\_front  npm install 或者 npm i  npm run dev |

或者是在idea中点击package.json里面的绿色箭头运行；

## Vue项目的开发方式

开发vue项目有两种方式：

1. 传统方式，在html页面中加入<script src=”../js/vue.min.js”>进行开发；（项目开发**很少**使用）

2、工程化方式，采用脚手架工具创建一个vue工程，然后进行开发；（项目开发**广泛**使用）

## Vue项目开发

.vue结尾的文件就是vue页面，我们也把它称为vue组件；

Vue页面（组件）一般是3个部分：

1. 写一个标签<template>，里面写html页面要展示的内容；
2. 写一个标签<script>，里面写javascript代码；
3. 写一个标签<style>，里面写css样式；

## Vue工程运行流程分析

vue项目的运行入口是：main.js

import App from './App.vue' *//从一个单文件组件中导入根组件*

const app = createApp(App);*//创建一个vue对象*

app.mount('#app') *//挂载到页面的#app这个id下*

#app是项目的首页index.html页面的一个标签元素的id；

## 登录页面

vue -- controller - service -- manager - mapper ( mysql、redis)



页面使用了element-plus进行页面开发；

## Element plus

饿了么团队开发并开源的；它非常适合于开发后台系统；

基于Vue2.x （element-ui） <https://element.eleme.cn/>

基于Vue3.x （element-plus） <http://element-plus.gitee.io/>

我们的项目，需要使用到一些页面的效果（表单、输入框、表格、按钮、布局、图标等等），我们采用的都是element-plus给我们提供的；

1、项目中要安装element-plus：

npm install element-plus --save

npm i element-plus --save

-g 全局安装，安装到D:\course\tool\node-v20.9.0-win-x64\node\_global

--save表示安装包信息会写入package.json的dependencies中，在dependencies中，那么项目打包就会依赖到该模块，如果项目打包时不需要依赖该模块，那么就使用--save-dev，它会在devDependencies下，表示项目开发需要依赖该模块，项目打包发布就不需要它；

2、使用element-plus

在main.js中

import ElementPlus from 'element-plus'

import 'element-plus/dist/index.css'

app.use(ElementPlus)

element-plus提供的组件：

1、Basic 基础组件

2、配置组件

3、Form 表单组件

4、Data 数据展示

5、Navigation 导航

6、Feedback 反馈组件

7、Others 其他

## 编写登录页面

LoginView.vue

左右结构；40% ：60%

## 登录

（1）账号字段：loginAct （account）登录账号；

（2）密码字段：loginPwd （password）登录密码；

（3）是否记住我字段：rememberMe

需求：前端非空验证；

1. 在<el-form>里面定义一个:rules="rules"；
2. 在要验证的el-input上的<el-form-item>中定义prop=xx

3、给要验证的字段定义验证规则；

## 安装axios依赖

官网：<https://axios-http.com/>

它是一个发送ajax请求的js库；

Vue中我们一般都是采用axios发送异步ajax请求；

npm install axios --save (两个杠)

## 使用axios

主要就是封装4个方法；

doGet() 查询

doPost() 新增

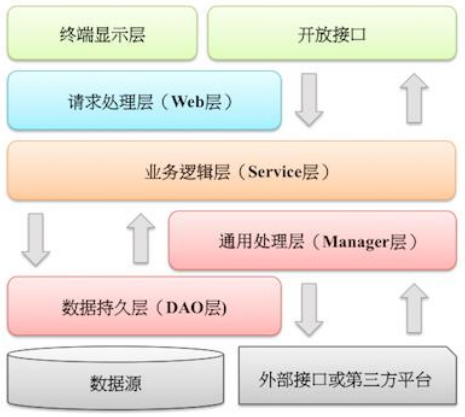
doPut() 修改

doDelete() 删除

## 后端项目搭建

创建dlyk-server的spring boot项目；

## 项目分层（后端项目）



## 领域对象划分

【参考】分层领域模型规约：

• DO（Data Object）：此对象与数据库表结构一一对应，通过 DAO 层向上传输数据源对象。

• DTO（Data Transfer Object）：数据传输对象，Service 或 Manager 向外传输的对象。

• BO（Business Object）：业务对象，可以由 Service 层输出的封装业务逻辑的对象。

• Query：数据查询对象，各层接收上层的查询请求。注意超过 2 个参数的查询封装，禁止使用 Map 类来传输。

• VO（View Object）：显示层对象，通常是Web向模板渲染引擎层传输的对象。

我们只使用其中的三个，如果都使用的话，反而把代码弄得复杂繁琐一些了；

1、DO（等价于我们的model）

2、Query （controller --> service -->manager --> dao 上一层往下一层传参数）

3、VO（等价于我们的result中的R对象）

## 跨域处理

跨域报错：Access to XMLHttpRequest at 'http://localhost:8089/api/login' from origin 'http://localhost:8081' has been blocked by CORS policy；

跨域：

源头：<http://localhost:8081> 目的地：[http://localhost:8089/api](http://localhost:8080/api)/login

1、协议不同， <http://localhost:8080> [https://localhost:8080](http://localhost:8080)

2、域名不同， <http://localhost:8080> [http://dlyk.bjpowernode.com:8080](http://localhost:8080)

3、端口不同 <http://localhost:8080> [http://localhost:8082](http://localhost:8080)

三个里面只要有任何一个不同就是跨域；

## axios请求的响应对象

axios请求后，返回的响应对象有6个字段：

1. **config**: {transitional: {…}, adapter: Array(2), transformRequest: Array(1), transformResponse: Array(1), timeout: 0, …}
2. **data**: {code: 200, msg: '成功', data: {…}}
3. **headers**: AxiosHeaders {cache-control: 'no-cache, no-store, max-age=0, must-revalidate', content-type: 'application/json;charset=UTF-8', expires: '0', pragma: 'no-cache'}
4. **request**: XMLHttpRequest {onreadystatechange: null, readyState: 4, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, …}
5. **status**: 200
6. **statusText**: ""

其中data就是后端springboot响应返回的那个R对象；

## 添加路由

路由：就是vue访问的路径；

比如: /

比如：/dashboard

1、用npm命令安装路由

npm install vue-router --save

2、在src下面新建一个router文件夹，在router文件夹下新建一个router.js文件，在router.js文件里面写入以下代码：

|  |
| --- |
| import { createRouter, createWebHistory } from "vue-router";  const router = createRouter({  history:createWebHistory(),  routes:[  {  path:'/',  component : () => import('@/view/DashboardView.vue'),  }  ]  })  export default router; |

3、在使用路由时导入路由：

在main.js中导入：

import router from './router/router.js'

1. 在main.js中使用路由

app.use(router);

5、使用<router-view/>标签渲染路由地址所对应的页面组件；

## 菜单加入图标

1、安装图标

npm install @element-plus/icons-vue --save

2、注册所有图标，在main.js中注册：

import \* as ElementPlusIconsVue from '@element-plus/icons-vue'

for (const [key, component] of Object.entries(ElementPlusIconsVue)) {

app.component(key, component)

}

3、使用图标，复制图标的代码，粘贴到项目中即可；

<el-icon><Plus /></el-icon>

<el-icon><Minus /></el-icon>

<el-icon><CirclePlus /></el-icon>

Vue中：

$refs 拿到页面上ref属性的那个dom元素；

$router vue路由对象，里面提供了一些方法供什么使用；

## 前后端登录的调试

1. vue前端通过axios提交loginAct、loginPwd参数；
2. 后端UsernamePasswordAuthenticationFilter接收；
3. 调用UserDetailsService的实现方法loadUserByUsername(String username)查询数据库；
4. 返回UserDetails对象给Spring Security框架进行密码比较，比较通过，登录成功；
5. 登录成功回调登录成功的handler，登录失败回调登录失败的handler；
6. 前端获取后端返回的json结果：

结果response对象包含6个字段：

（1）config

（2）data 这个里面才是后端返回的数据，所以我们取该字段就行了；

（3）headers

（4）request

（5）status

（6）statusText

## 跨域的解决

Spring Security的跨域解决；

/api/login 地址，该地址没有controller，是spring security框架提供的，所以需要从spring security中解决跨域问题；

## 密码加密

bcrypt加密，是目前最安全的加密方式，不容易被破解;

MD5加密，安全性比bcrypt差一些，更容易被破解；

## 管理面板主页面

DashboardView.vue

## 前后端分离登录状态保持

前后端分离的项目一般会使用token（jwt）实现登录状态的保持；（java web ： session）  
token其实就是一个随机字符串（字符串要求是唯一的，不同人的token都不能相同），当用户在登录页面输入账号和密码后，前端将账号密码发送给后端，后端检验完账号和密码后，会生成一个随机不重复的字符串即（token）,并将其响应给前端，前端拿到token后，需要在客户端进行持久化存储（一般会写在localStorage或者sessionStorage中），那么下次在向后端数据接口发送请求的时候，一般需要将token一并发送给后端数据接口，后端数据接口会对token进行校验，如果合法则正常响应请求，如果不合法，则提示未登录。

## sessionStorage与localStorage

它们是javascript对象，浏览器支持这两个对象，可以直接使用；

localStorage和sessionStorage都是用来在浏览器客户端存储临时信息的对象；

sessionStorage、localStorage区别？

sessionStorage只在一个浏览器页面有效，比如你打开一个新的tab浏览器页会失效，你关闭浏览器后，再打开浏览器也会失效；

localStorage在整个浏览器中都有效，重启浏览器也有效；除非你手动删除了localStorage，才会失效；

## 免登录（自动登录）

不需要登录，就直接进入系统；

<http://localhost:8081/> 访问斜杆，那么渲染的是LoginView.vue，在该页面中判断是否要免登录？

如果要免登录，则渲染LoginView.vue这个页面的时候，就自动请求后端完成登录；

免登录的条件：上一次登录，用户选择了“记住我”，则在7天内自动给用户登录，也就是用户不需要输入账号和密码去登录；

## 退出登录

用户在系统中做完了相关操作了，不需要再使用系统，他可以选择退出登录；

1、把后端redis中的登录token删除；

2、把前端sessionStorage/localStorage里面的登录token删除；

## token续期

选择了记住我，token有效期7天；

没有选择了记住我，token有效期30分钟；

在用户操作页面功能的过程中，要对token进行续期，也就是延长token失效时间；

就和java web的session，默认是30分钟；

5分钟我对页面不做任何，此时session的过期时间就变成了25分钟；

我现在过5分钟后点一下页面，此时要把session的过期时间再改成30分钟；

# 账户管理

这是以4s店为项目背景的客户营销管理系统，（4s店即汽车经销商），4s店里面有很多销售顾问，每个销售人员都可以登录该系统，每个销售人员都有一个账号；这个账号需要管理员创建和维护；

1. 所有人员的账号信息列表进行分页展示；（OK）
2. 查看某个账号的详情；（OK）
3. 新增创建一个新账号；（OK）
4. 修改某个人的账号；（修改他的账户名、状态（锁定、禁用.....））（OK）
5. 删除一个账号；（OK）
6. 批量删除；(OK)

## $route与$router

**$route**: 获取当前路由对象的属性信息；比如：

this.$route.path； 拿到路由路径

this.$route.params.id； 拿到路由地址中的id参数

**$router**: 调用当前路由对象的方法，比如：

this.$router.push(“/dashboard/user”)；//跳转到指定的路由地址；

this.$router.go(-1); //跳转到上一个路由地址

## 页面刷新

window.location.reload(); //原始的js的方式刷新，是整个页面都刷新；

我们想实现局部的页面组件进行刷新；

采用provide / inject 组合，实现页面组件局部刷新；

1、修改DashboardView.vue页面:

|  |
| --- |
| <template>  <router-view v-if="isRouterAlive"/>  </template> |
| export default {  data() {  return {  isRouterAlive : true,  }  },  provide() {  return {  reload: ()=> {  this.isRouterAlive = false;  this.$nextTick(function () {  this.isRouterAlive = true;  })  },  }  }, |

页面1 -- 页面2 -- 页面3 -- 页面4 -- 页面5

页面1的数据可以传给页面3、页面5... 都是可以的；

多级页面的数据穿透；（provide/inject组合就行了）

声明reload函数，具体实现是通过isRouterAlive等于true或false来控制router-view的显示或隐藏，从而控制页面的再次加载（达到刷新的目的）；

2、在需要进行刷新的页面中注入DashboardView.vue组件提供（provide）的 reload函数，然后直接用this.reload()来调用进行刷新；

3、$nextTick()函数表示vue执行完渲染后会执行this.nextTick()里面的callback函数；

## 数据权限

管理员可以看到所有用户的信息；

其他人员只能看到自己的信息；

数据权限：sql语句进行过滤，管理员登录查询所有数据，普通用户登录就只查询自己的数据；

# 市场活动

市场活动就是为了汽车销售进行的一些销售活动，比如：在一些网站投放广告、抖音上直播、线下车展、零首付活动、以旧换新活动.......，目的是为了销售汽车，提高销售额，获取更大市场份额，让更多消费者认识xx品牌，提高知名度等；

1. 市场活动数据分页展示; （OK）
2. 市场活动的搜索；（OK）
3. 录入市场活动；（OK）
4. 编辑市场活动；（OK）
5. 查看市场活动详情；（OK）
6. 市场活动备注（包括备注信息的编辑、删除）；（OK）
7. 删除市场活动；
8. 批量删除市场活动；

## element-plus的国际化（中文）

1、在main.js中导入国际化的中文包

import zhCn from 'element-plus/dist/locale/zh-cn.mjs'

2、在app对象使用ElementPlus的时候，要指定使用的语言环境

app.use(ElementPlus, {

locale: zhCn,

})

## 前后端参数传递与接收

get 和 delete请求传参数是一样；

post 和 put请求传参数是一样；

1. formData往后端传参数，

前端是get、delete、post、put请求，后端接收：

1. @RequestParam
2. request.getParamter()
3. UserQuery对象把参数一个个映射到对象中
4. {id : 1208, name : ‘张三’， phone : ‘13720202020’}

前端是get、delete请求，后端接收：

1. @RequestParam
2. request.getParamter()
3. UserQuery对象把参数一个个映射到对象中

前端是post、put请求，后端接收：

@RequestBody注解接收，因为前端传过来的是json格式

1. /api/user/12056

前端是get、delete、post、put请求，后端接收：

@PathVariable()注解接收参数

1. 在请求头传的参数，header里面放了一个参数；

前端是get、delete、post、put请求，后端接收：

@RequestHeader()注解接收

request.getHeader()方法接收

1. 前端传过来的是文件

接收：MultipartFile

# 线索管理

线索：就是有一定意向的客户的联系方式，（有购买需求，不一定会买），拿到这些意向客户的联系方式（手机号、微信.....）把这些有需求的客户信息拿到，录入到系统里面，后续跟踪这些线索(联系他，打电话、微信)，促使交易（让他来买我们的产品，他能买车）；有的线索最后可能不会成交，有的线索会成交；

这些线索（客户的联系方式）通过各种各样的途径获得：线下车展、抖音直播、投放广告、客户上门来看车......；

1、线索数据的分页列表；

2、新增线索（录入线索）；

3、编辑线索信息；

4、查看线索详情；

5、线索填写跟踪记录；  
6、展示线索的历史跟踪记录（分页）；

7、线索的历史跟踪记录编辑

8、线索的历史跟踪记录删除（逻辑删除）

9、线索转换为客户

10、删除线索；

11、批量删除线索；

## EasyExcel

官网：<https://easyexcel.opensource.alibaba.com/>

EasyExcel是一个基于Java的、快速、简洁、解决大文件内存溢出的Excel处理工具（jar包）。

他能让你在不用考虑性能、内存的等因素的情况下，快速完成Excel的读、写等功能。

原来操作Excel使用apache poi和jxl项目，这两个项目：

1. 代码开发繁琐一些；
2. 大文件解析会产生内存溢出；（2G的Excel文件需要导入/导出）

EasyExcel完美解决了上面的两个问题；

|  |
| --- |
| <dependency>  <groupId>com.alibaba</groupId>  <artifactId>easyexcel</artifactId>  <version>3.3.3</version>  </dependency> |

# 客户管理

动力云客的主要管理对象是客户，客户怎么来的？是通过各种渠道拿到广大消费者的联系方式（线索），线索通过一定的筛选（销售人员去跟踪这些消费者），对有比较强的意向的消费者就转换为客户，转为客户后，再通过对客户进行跟踪沟通，最终促成交易；

市场活动 / 各种渠道 --> 线索 --> 客户 --> 交易(商机) --> 成交

1、客户数据的分页列表；

2、批量导出客户信息到Excel；

3、选择导出客户信息到Excel；

4、查看客户详情；

5、客户详情页填写跟踪记录；

6、客户详情页创建交易；

# 交易管理

交易也叫商机，交易管理是客户通过前期的跟踪沟通，有非常明确的购买意向客户，给客户提供了详细的报价，客户索要一定的优惠等等都进行了充分沟通，客户可以交纳基础的定金，我们就会为这些客户创建交易，由于购买是的大件商品，价格也不低，所以成交过程会花费一定的时间，交定金到付全款，到交付产品，会经历一些阶段，在交易的过程中，销售人员依然需要与客户保持联系沟通，直到最终成交；

1. 交易（商机）分页列表展示；
2. 交易详情页；
3. 填写交易跟踪记录；
4. 历史跟踪记录展示；
5. 更新交易阶段；
6. 历史交易记录（阶段）展示；

# 数据统计

数据统计是后端系统中常见的功能需求，每天、每月、每季度、每年能统计出系统的相关业务数据，便于及时总结复盘，发现问题，改进提升；

## 概览统计

1、市场活动

12/3163

目前在进行的市场活动数量 / 数据库中所有的市场活动总数

2、线索总数

172,000

线索表有多少条数据

3、客户总数

562

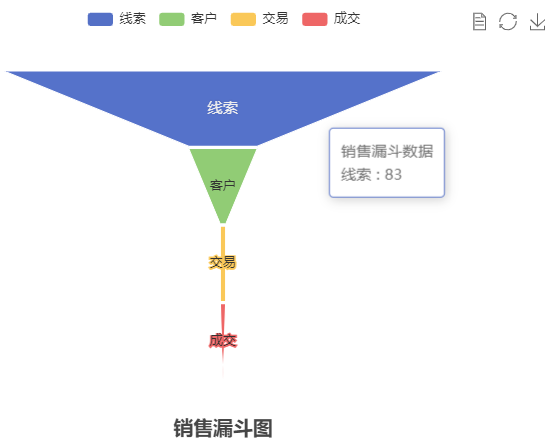
客户表总共有多少条数据

4、交易总额

12088/268500

成功的交易额 / 总的交易额（包含成功和不成功的）

## 销售漏斗图



## 线索来源统计饼图

## 市场活动每月统计柱状图

## 线索数据每天统计柱状图

## 客户数据每天统计柱状图

## 交易数据每天统计柱状图

# ECharts图表库

漏斗图

饼图

柱状图

<https://echarts.apache.org/>

一个基于 JavaScript 的开源可视化图表库；

原来是百度公司开发并且开源；

使用步骤：

1、从 npm 获取安装：

npm install echarts --save

2、看文档，看参考示例进行开发；https://echarts.apache.org/examples/zh/index.html

（1）文档: (使用手册、API、配置项手册)

（2）示例 （最主要参考它提供的示例进行开发）

Vue项目上线部署主要采用Nginx服务器；

# Nginx部署Vue项目

## Nginx服务器介绍

1、Nginx 是一个高性能的web服务器；（服务器）

2、Nginx 是由俄罗斯人 Igor Sysoev 采用C语言开发编写的，第一个公开版本0.1.0发布于2004年10月4日；

Nginx的作者；



3、Nginx 特点是占有内存少，并发处理能力强，以高性能、低系统资源消耗而闻名，在全世界范围内大量的网站使用了Nginx，国内互联网中也大量使用了Nginx，比如：淘宝、新浪、搜狐、网易、美团等；

7、Nginx是免费开源的，同时Nginx也有收费的商业版本，商业版本提供了性能优化、宕机等紧急问题处理等技术支持和服务；

商业版本：<http://www.nginx.com>

开源版：<http://nginx.org>

## Nginx环境搭建

### 下载

免费开源版的官方网站：<http://nginx.org>

Nginx 有 windows 版本和 linux 版本，但推荐在 linux 下使用 Nginx；

### 安装

#### 安装前准备

Nginx的安装需要Linux安装相关的几个库，否则配置和编译会出现错误；

执行如下命令：

yum install gcc openssl openssl-devel pcre pcre-devel zlib zlib-devel -y

#### 正式安装

1、解压下载下来的nginx文件，执行命令：tar -zxvf nginx-1.24.0.tar.gz

2、切换至解压后的nginx主目录，执行命令：cd nginx-1.24.0

3、在nginx主目录nginx-1.24.0下执行命令：./configure --prefix=/opt/nginx   
（其中--prefix是指定nginx安装路径）

4、执行命令进行编译：make

5、执行命令进行安装：make install

## Nginx启动

1、启动nginx执行命令：

/opt/nginx/sbin/nginx -c /opt/nginx/conf/nginx.conf

（其中-c是指定配置文件）

2、检查Nginx是否启动：通过查看进程：ps -ef | grep nginx

## Nginx重启

/opt/nginx/sbin/nginx -s reload

## Nginx关闭

Nginx的关闭有多种方式：

1、优雅关闭Nginx：

找出nginx的进程号：ps -ef | grep nginx

执行命令：kill -QUIT 主pid kill -QUIT 4702

其中pid是主进程号的pid（master process），其他为子进程pid（worker process）

优雅关闭会将已经进来的请求处理完后再关闭；

2、快速关闭Nginx：

找出nginx的进程号：ps -ef | grep nginx

kill -TERM 主pid

快速关闭会直接关闭，已经进来的请求也不会处理；(暴力方式)

## Nginx配置检查

**conf/nginx.conf**

当修改Nginx配置文件后，可以使用Nginx命令进行配置文件语法检查，用于检查Nginx配置文件是否正确；

检查Nginx配置文件是否正确：  
/opt/nginx/sbin/nginx -c /opt/nginx/conf/nginx.conf -t

## Nginx部署Vue项目

1. 把vue项目打包；

修改Nginx配置文件nginx.conf：

location / {

root html/dlyk;

try\_files $uri $uri/ /index.html;

index index.html index.htm;

}

红色部分就是重点配置；