1 学习目标

- 使用资料搭建后台系统
- 会使用nginx进行反向代理
- 实现商品分类CRUD功能
- 掌握cors解决跨域

2 使用域名访问本地项目

2.1 统一环境

我们现在访问页面使用的是：http://localhost:9001

有没有什么问题？

实际开发中，会有不同的环境：

- 开发环境：自己的电脑
- 测试环境：提供给测试人员使用的环境
- 预发布环境：数据是和生成环境的数据一致，运行最新的项目代码进去测试
- 生产环境：项目最终发布上线的环境

如果不同环境使用不同的ip去访问，可能会出现一些问题。为了保证所有环境的一致，我们会在各种环境下都使用域名来访问。

我们将使用以下域名：

- 主域名是：www.mrshop.com，
- 管理系统域名：manage.mrshop.com
- 网关域名：api.mrshop.com
- ...

是最终，我们希望这些域名指向的还是我们本机的某个端口。

那么，当我们在浏览器输入一个域名时，浏览器是如何找到对应服务的ip和端口的呢？

2.2 域名解析

一个域名一定会被解析为一个或多个ip。这一般会包含两步：

- 本地域名解析

  浏览器会首先在本机的hosts文件中查找域名映射的IP地址，如果查找到就返回IP ，没找到则进行域名服务器解析，一般本地解析都会失败，因为默认这个文件是空的。

  - Windows下的hosts文件地址：C:/Windows/System32/drivers/etc/hosts
  - Linux下的hosts文件所在路径： /etc/hosts

样式：

```
# My hosts
127.0.0.1 localhost
0.0.0.0 account.jetbrains.com
127.0.0.1 www.xmind.net
```

- 域名服务器解析

  本地解析失败，才会进行域名服务器解析，域名服务器就是网络中的一台计算机，里面记录了所有注册备案的域名和ip映射关系，一般只要域名是正确的，并且备案通过，一定能找到。

2.3 解决域名解析问题

我们不可能去购买一个域名，因此我们可以伪造本地的hosts文件，实现对域名的解析。修改本地的host为：

```
127.0.0.1 api.mrshop.com
127.0.0.1 manage.mrshop.com
```

这样就实现了域名的关系映射了。

注意:有可能出现不能修改文件的问题

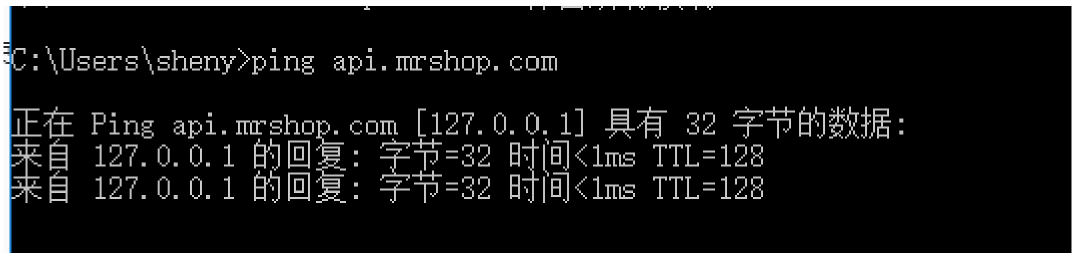
1:用户权限的问题

2:文件有只读属性

添加了两个映射关系：

- 127.0.0.1 api.mrshop.com ： 我们的网关Zuul
- 127.0.0.1 manage.shop.com：我们的后台系统地址

现在，ping一下域名试试是否畅通：



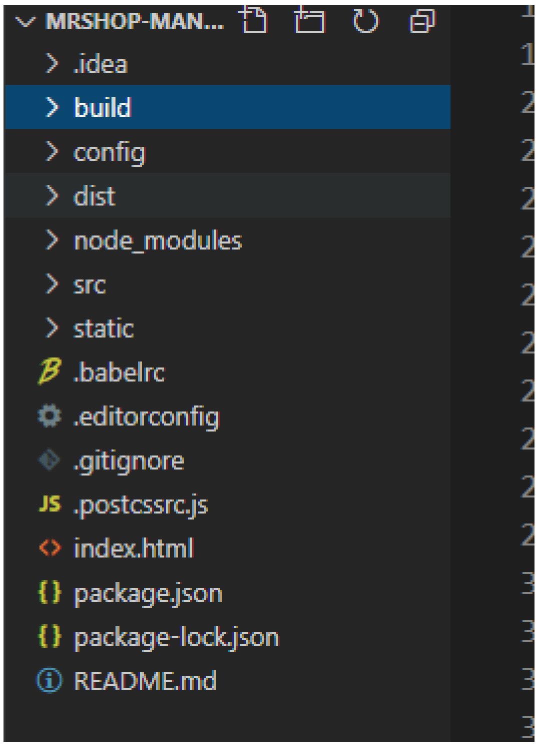
```
C:\Users\sheny>ping api.mrshop.com

正在 Ping api.mrshop.com [127.0.0.1] 具有 32 字节的数据:
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
```

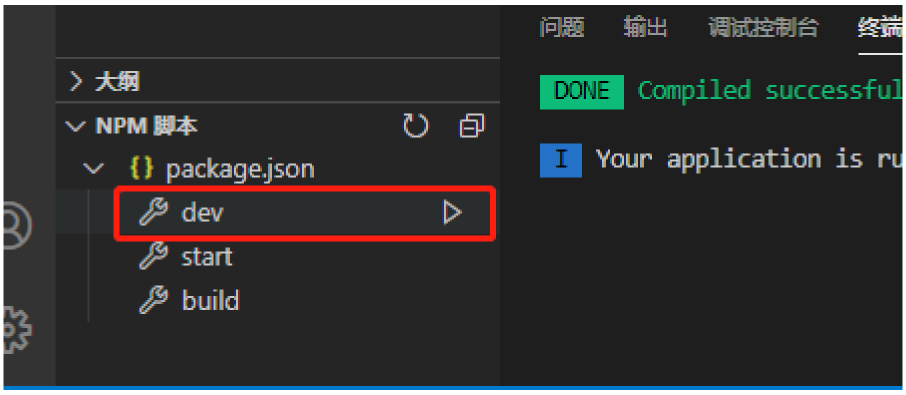
2.3.1 测试

启动后端VUE项目

解压 mrshop-manage-web.rar到vscode工作空间

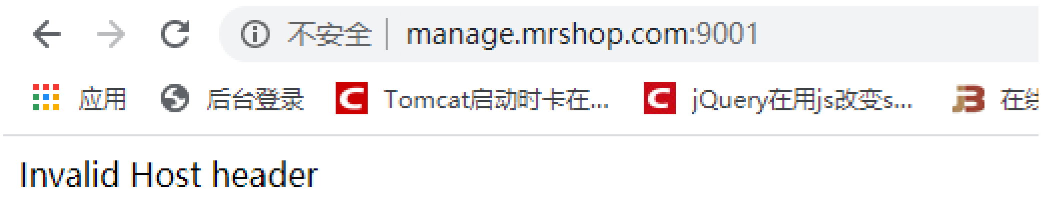
使用vscode 打开项目





```
DONE  Compiled successful
I  Your application is ru
```

编译项目.....



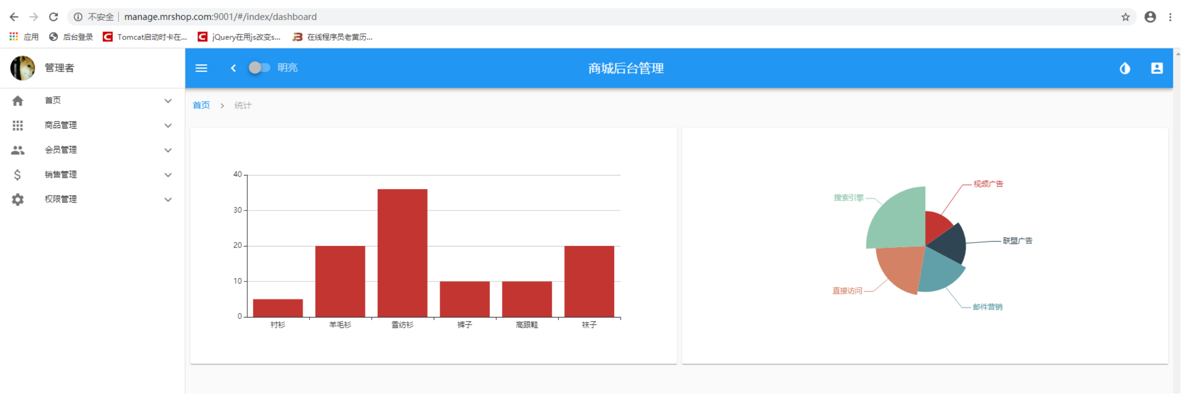
Invalid Host header

输入域名 + 9001端口号出现上述错误

打开build文件夹下webpack.dev.conf.js文件在devServer下加入

```
disableHostCheck: true,
// these devServer options should be customized in /config/index
devServer: {
  disableHostCheck: true,
  clientLogLevel: 'warning',
  historyApiFallback: {
    rewrites: [
      { from: /.*/, to: path.posix.join(config.dev.assetsPublicP
```

再次编译项目



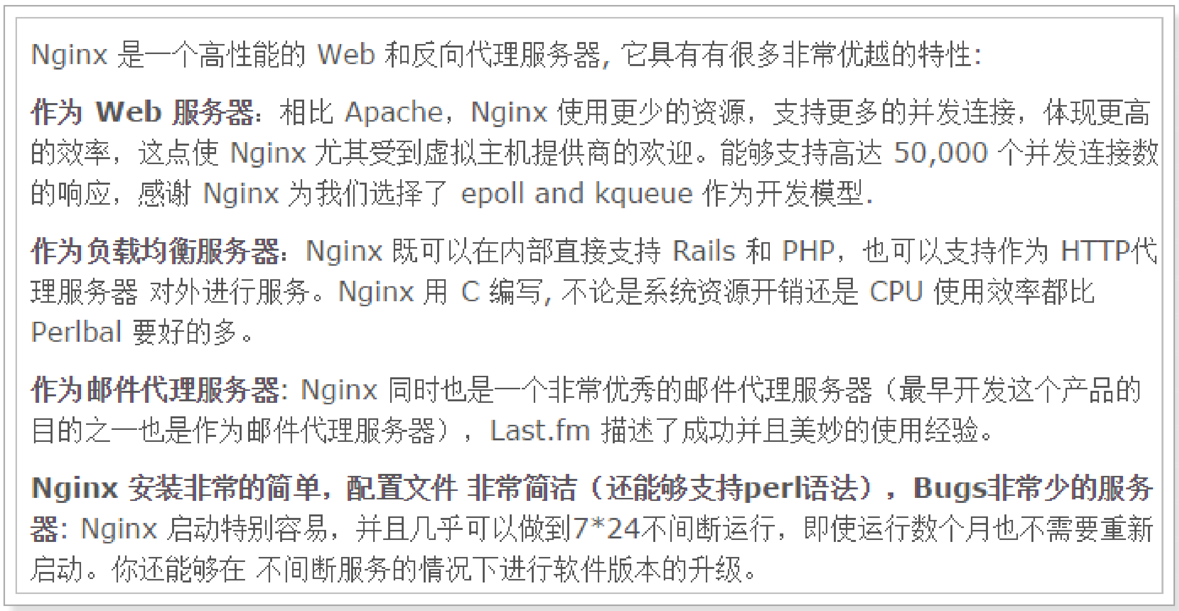
域名解析成功

2.4 nginx解决端口问题

我们希望的是直接域名访问：`http://manage.mrshop.com`。这种情况下端口默认是80，如何才能把请求转移到9001端口呢？

这里就要用到反向代理工具：Nginx

2.4.1 什么是Nginx



Nginx 是一个高性能的 Web 和反向代理服务器, 它具有有很多非常优越的特性:

作为 Web 服务器：相比 Apache，Nginx 使用更少的资源，支持更多的并发连接，体现更高的效率，这点使 Nginx 尤其受到虚拟主机提供商的欢迎。能够支持高达 50,000 个并发连接数的响应，感谢 Nginx 为我们选择了 epoll and kqueue 作为开发模型.

作为负载均衡服务器：Nginx 既可以在内部直接支持 Rails 和 PHP，也可以支持作为 HTTP代理服务器 对外进行服务。Nginx 用 C 编写, 不论是系统资源开销还是 CPU 使用效率都比 Perlbal 要好的多。

作为邮件代理服务器: Nginx 同时也是一个非常优秀的邮件代理服务器（最早开发这个产品的目的之一也是作为邮件代理服务器），Last.fm 描述了成功并且美妙的使用经验。

Nginx 安装非常的简单，配置文件 非常简洁（还能够支持perl语法），Bugs非常少的服务器: Nginx 启动特别容易，并且几乎可以做到7*24不间断运行，即使运行数个月也不需要重新启动。你还能够在 不间断服务的情况下进行软件版本的升级。

NIO：not-blocking-io 非阻塞IO

BIO：blocking-IO 阻塞IO

nginx可以作为web服务器，但更多的时候，我们把它作为网关，因为它具备网关必备的功能：

- 反向代理

- 负载均衡
- 动态路由
- 请求过滤

2.4.2 nginx作为web服务器

Web服务器分2类:

- web应用服务器，如:
    - tomcat
    - resin
    - jetty
- web服务器，如:
    - Apache 服务器
    - Nginx
    - IIS

区分：web服务器不能解析jsp等页面，只能处理js、css、html等静态资源。
并发：web服务器的并发能力远高于web应用服务器。

Nginx + tomcat !!!! 负载均衡

2.4.3 nginx作为反向代理

什么是反向代理?

- 代理：通过客户机的配置，实现让一台服务器代理客户机，客户的所有请求都交给代理服务器处理。
- 反向代理：用一台服务器，代理真实服务器，用户访问时，不再是访问真实服务器，而是代理服务器。

nginx可以当做反向代理服务器来使用:

- 我们需要提前在nginx中配置好反向代理的规则，不同的请求，交给不同的真实服务器处理
- 当请求到达nginx，nginx会根据已经定义的规则进行请求的转发，从而实现路由功能

2.4.4 安装和使用

2.4.4.1 安装

解压nginx-1.16.1.zip

安装完成

2.4.4.2 使用

nginx可以通过命令行来启动，操作命令:

- 启动: `start nginx.exe`
- 停止: `nginx.exe -s stop`
- 重新加载: `nginx.exe -s reload`

反向代理配置

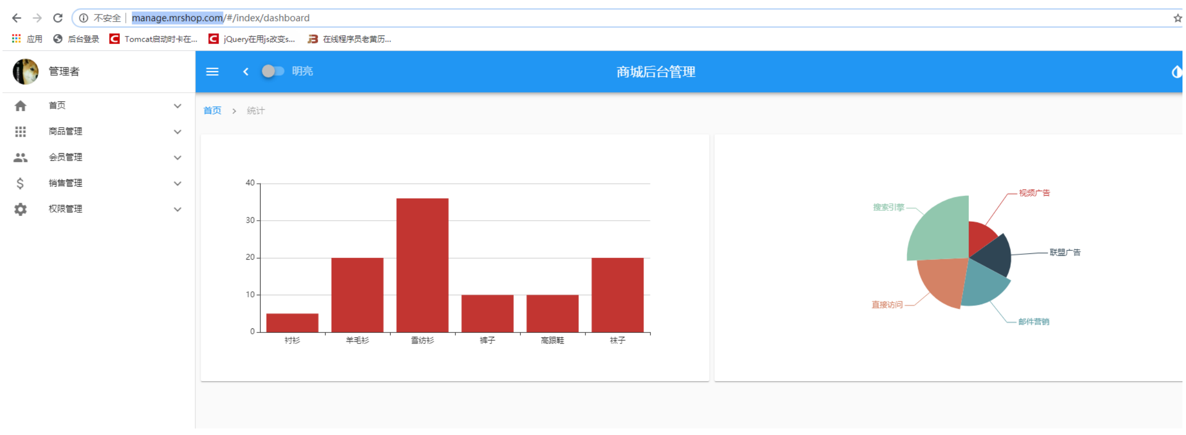
修改nginx/conf/nginx.conf文件

```
#user  nobody;
```

```
worker_processes  1;

#error_log  logs/error.log;
#error_log  logs/error.log  notice;
#error_log  logs/error.log  info;

#pid        logs/nginx.pid;


events {
    worker_connections  1024;
}


http {
    include       mime.types;
    default_type  application/octet-stream;

    sendfile        on;
    keepalive_timeout  65;

    server {
        listen       80;
        server_name  manage.mrshop.com;

        proxy_set_header X-Forwarded-Host $host;
        proxy_set_header X-Forwarded-Server $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;


        location / {
            proxy_pass http://127.0.0.1:9001;
            proxy_connect_timeout 600;
            proxy_read_timeout 600;
        }

    }

    server {
        listen       80;
        server_name  api.mrshop.com;

        proxy_set_header X-Forwarded-Host $host;
        proxy_set_header X-Forwarded-Server $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;


        location / {
            proxy_pass http://127.0.0.1:8088;
            proxy_connect_timeout 600;
            proxy_read_timeout 600;
        }

    }


}
```

2.4.4.3 测试

浏览器输入http://manage.mrshop.com/



出现上图效果即可

现在实现了域名访问网站了，中间的流程是怎样的呢?

1. 浏览器准备发起请求，访问http://mamage.mrshop.com，但需要进行域名解析
2. 优先进行本地域名解析，因为我们修改了hosts，所以解析成功，得到地址：127.0.0.1
3. 请求被发往解析得到的ip，并且默认使用80端口：http://127.0.0.1:80

   本机的nginx一直监听80端口，因此捕获这个请求
4. nginx中配置了反向代理规则，将manage.mrshop.com代理到127.0.0.1:9001，因此请求被转发
5. 后台系统的webpack server监听的端口是9001，得到请求并处理，完成后将响应返回到nginx
6. nginx将得到的结果返回到浏览器

3 商品分类管理(查询)

3.1 mingrui-shop-common-core工程

3.1.1 在mingrui-shop--commons项目下新建mingrui-shop-common-core项目

3.1.2 pom.xml

```
        <!--处理json与各种数据类型或文档类型的转换-->
        <dependency>
            <groupId>com.google.code.gson</groupId>
            <artifactId>gson</artifactId>
            <version>2.8.5</version>
        </dependency>
        <!--json对象序列化和反序列化的支持-->
        <dependency>
            <groupId>org.codehaus.jackson</groupId>
            <artifactId>jackson-core-lgpl</artifactId>
            <version>1.9.13</version>
        </dependency>
        <dependency>
            <groupId>org.codehaus.jackson</groupId>
```

```xml
            <artifactId>jackson-mapper-lgpl</artifactId>
            <version>1.9.13</version>
        </dependency>
        <!--java对象和json对象之间的转换-->
        <dependency>
            <groupId>org.codehaus.jackson</groupId>
            <artifactId>jackson-core-asl</artifactId>
            <version>1.9.13</version>
        </dependency>
        <dependency>
            <groupId>org.codehaus.jackson</groupId>
            <artifactId>jackson-mapper-asl</artifactId>
            <version>1.9.13</version>
        </dependency>
        <!--alibaba的json处理工具-->
        <dependency>
            <groupId>com.alibaba</groupId>
            <artifactId>fastjson</artifactId>
            <version>1.2.62</version>
        </dependency>
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-databind</artifactId>
            <version>2.11.2</version>
        </dependency>
```

3.1.3 新建包com.baidu.shop.utils

3.1.4 JSONUtil

```java
import com.alibaba.fastjson.JSONObject;
import org.codehaus.jackson.JsonParseException;
import org.codehaus.jackson.map.JsonMappingException;
import org.codehaus.jackson.map.ObjectMapper;
import org.codehaus.jackson.type.TypeReference;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.google.gson.JsonArray;
import com.google.gson.JsonElement;
import com.google.gson.JsonObject;
import com.google.gson.JsonParser;
import com.google.gson.reflect.TypeToken;

import java.io.FileOutputStream;
import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
/**
 * @ClassName JSONUtil
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
```

```java
public class JSONUtil {
    private static Gson gson = null;

    static {
        gson = new GsonBuilder().setDateFormat("yyyy-MM-dd
HH:mm:ss").create();// todo yyyy-MM-dd HH:mm:ss
    }

    public static synchronized Gson newInstance() {
        if (gson == null) {
            gson = new GsonBuilder().setDateFormat("yyyy-MM-dd
HH:mm:ss").create();
        }
        return gson;
    }

    public static String toJsonString(Object obj) {
        return gson.toJson(obj);
    }

    public static <T> T toBean(String json, Class<T> clz) {

        return gson.fromJson(json, clz);
    }

    public static <T> Map<String, T> toMap(String json, Class<T> clz) {
        Map<String, JsonObject> map = gson.fromJson(json, new
TypeToken<Map<String, JsonObject>>() {
        }.getType());
        Map<String, T> result = new HashMap<String, T>();
        for (String key : map.keySet()) {
            result.put(key, gson.fromJson(map.get(key), clz));
        }
        return result;
    }

    public static Map<String, Object> toMap(String json) {
        Map<String, Object> map = gson.fromJson(json, new TypeToken<Map<String,
Object>>() {
        }.getType());
        return map;
    }

    public static <T> List<T> toList(String json, Class<T> clz) {
        JsonArray array = new JsonParser().parse(json).getAsJsonArray();
        List<T> list = new ArrayList<T>();
        for (final JsonElement elem : array) {
            list.add(gson.fromJson(elem, clz));
        }
        return list;
    }

    /**
     * 从json字符串中获取需要的值
     *
     * @param json
     * @param clazz 要转换的类型
     * @return
     */
    public static <T> Object getObjectByKey(String json, Class<T> clazz) {
        if (json != null && !"".equals(json)) {
            return JSONObject.parseObject(json, clazz);
        }
        return null;
    }

    /**
     * 从json字符串中获取需要的值
     *
     * @param json
     * @param clazz 要转换的类型
     * @return
     */
    public static <T> List<T> getListByKey(String json, Class<T> clazz) {
        if (json != null && !"".equals(json)) {
            return JSONObject.parseArray(json, clazz);
        }
        return null;
    }

    /**
     * 从json字符串中获取需要的值
     *
     * @param json
     * @param key
     *            键
     * @return
     */
    public static String getStrByKey(String json, String key) {
        String str = "";
        if (json != null && !"".equals(json)) {
            JSONObject j = JSONObject.parseObject(json);
            if (j.get(key) != null) {
                str = j.get(key).toString();
            }
        }
        return str;
    }

    /**
     * 向文件中写数据
     *
     * @param _sDestFile
     * @param _sContent
     * @throws IOException
     */
    public static void writeByFileOutputStream(String _sDestFile, String
_sContent) throws IOException {
        FileOutputStream fos = null;
        try {
            fos = new FileOutputStream(_sDestFile);
            fos.write(_sContent.getBytes());
        } catch (Exception ex) {
            ex.printStackTrace();
        } finally {
            if (fos != null) {
                fos.close();
                fos = null;
            }

        }

    }

    /**
     * 非空
     *
     * @param str
     * @return true:不为空 false：空
     */
    public static boolean noEmpty(String str) {
        boolean flag = true;
        if ("".equals(str)) {
            flag = false;
        }
        return flag;
    }

    /**
     * 将"%"去掉
     *
     * @param str
     * @return
     */
    public static double getDecimalByPercentage(String str) {
        double fuse = 0.0;
        if (!"".equals(str) && str != null) {
            if (str.split("%").length > 0) {
                fuse = Double.parseDouble(str.split("%")[0]);
                return fuse;
            }
        }
        return 0.0;
    }

    /**
     * 保留2位小数
     *
     * @param number
     * @return
     */
    public static double ConversionFraction(double number) {
        return Math.floor(number * 100 + 0.5) / 100;
    }

    public static float ConversionM(double number) {
        return (float) JSONUtil.ConversionFraction(number / 1024 / 1024);
    }

    public static String getErrorText(String s) {
        JSONObject j = JSONObject.parseObject(s);
        return
j.getJSONObject(j.keySet().iterator().next()).get("errortext").toString();
    }
    public static String getSingleJobId(String s) throws Exception {
        JSONObject j = JSONObject.parseObject(s);
        try {
            return
j.getJSONObject(j.keySet().iterator().next()).get("jobid").toString();
        } catch (Exception e) {
            try {
                return
j.getJSONObject(j.keySet().iterator().next()).get("errortext").toString();
            } catch (Exception e1) {
                throw new Exception(e1.getMessage());

            }

        }
    }

    public static <T> T readValue(String jsonStr, TypeReference type)
            throws JsonParseException, JsonMappingException, IOException {
        ObjectMapper mapper = new ObjectMapper();
        return mapper.readValue(jsonStr, type);
    }

    public static JSON_TYPE getJSONType(String str) {
        if (null == str || "".equals(str)) {
            return JSON_TYPE.JSON_TYPE_ERROR;
        }

        final char[] strChar = str.substring(0, 1).toCharArray();
        final char firstChar = strChar[0];

        if (firstChar == '{') {
            return JSON_TYPE.JSON_TYPE_OBJECT;
        } else if (firstChar == '[') {
            return JSON_TYPE.JSON_TYPE_ARRAY;
        } else {
            return JSON_TYPE.JSON_TYPE_ERROR;
        }
    }

    public enum JSON_TYPE {
        /** JSONObject */
        JSON_TYPE_OBJECT,
        /** JSONArray */
        JSON_TYPE_ARRAY,
        /** 不是JSON格式的字符串 */
        JSON_TYPE_ERROR
    }
}
```

3.1.5 新建包com.baidu.shop.status

3.1.6 HTTPStatus

```
/**
```

```
 * @ClassName HTTPStatus
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
public class HTTPStatus {

    public static final int OK = 200;//成功

    public static final int ERROR = 500;//失败

}
```

3.1.7 新建包com.baidu.shop.base

3.1.8 Result

> 泛型的高级使用 : http://note.youdao.com/noteshare?id=a83103814321b63e12f932c5552c197d&sub=F231BBF10BC54EBF8EF8AB87967A153D
>
> lombok : https://www.jianshu.com/p/2543c71a8e45

```
import lombok.Data;
import lombok.NoArgsConstructor;

/**
 * @ClassName Result
 * @Description: 统一返回
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@Data
@NoArgsConstructor
public class Result<T> {

    private Integer code;//返回码

    private String message;//返回消息

    private T data;//返回数据

    public Result(Integer code, String message, Object data) {
        this.code = code;
        this.message = message;
        this.data = (T) data;
    }
}
```

BaseApiService

```
import com.baidu.shop.status.HTTPStatus;
import com.baidu.shop.utils.JSONUtil;
```

```java
import lombok.Data;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Component;

/**
 * @ClassName BaseApiService
 * @Description: 接口的实现类继承此类
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@Data
@Component
@Slf4j
public class BaseApiService<T> {

    public Result<T> setResultError(Integer code, String msg) {
        return setResult(code, msg, null);
    }

    // 返回错误，可以传msg
    public Result<T> setResultError(String msg) {

        return setResult(HTTPStatus.ERROR, msg, null);
    }

    // 返回成功，可以传data值
    public Result<T> setResultSuccess(T data) {
        return setResult(HTTPStatus.OK, HTTPStatus.OK + "", data);
    }

    // 返回成功，沒有data值
    public Result<T> setResultSuccess() {
        return setResult(HTTPStatus.OK, HTTPStatus.OK + "", null);
    }

    // 返回成功，沒有data值
    public Result<T> setResultSuccess(String msg) {
        return setResult(HTTPStatus.OK, msg, null);
    }

    // 通用封装
    public Result<T> setResult(Integer code, String msg, T data) {
        log.info(String.format("{code : %s , msg : %s , data : %s}",code,msg,
JSONUtil.toJsonString(data)));
        return new Result<T>(code, msg, data);
    }

}
```

3.2 service-api工程

3.2.1 mingrui-shop-service-api/pom.xml

```xml
        <!-- SpringBoot-整合Web组件 -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <!--Entity 中的@Table 和@Id需要次注解-->
        <dependency>
            <groupId>javax.persistence</groupId>
            <artifactId>persistence-api</artifactId>
            <version>1.0.2</version>
        </dependency>

         <!--2.3版本之后web删除了验证插件-->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-validation</artifactId>
        </dependency>

        <!--引入common工程代码-->
        <dependency>
            <groupId>com.baidu</groupId>
            <artifactId>mingrui-shop-common-core</artifactId>
            <version>1.0-SNAPSHOT</version>
        </dependency>
```

3.2.2 在mingrui-shop-service-api项目下新建mingrui-shop-service-api-xxx项目

3.2.3 pom.xml

```
        <!--帮助开发人员快速生成API文档-->
        <dependency>
            <groupId>io.springfox</groupId>
            <artifactId>springfox-swagger2</artifactId>
            <version>2.9.2</version>
        </dependency>
        <!--提供可视化的API文档-->
        <dependency>
            <groupId>io.springfox</groupId>
            <artifactId>springfox-swagger-ui</artifactId>
            <version>2.9.2</version>
        </dependency>
```

3.2.4 新建包com.baidu.shop.entity

3.2.5 CategoryEntity

```
import io.swagger.annotations.ApiModel;
import io.swagger.annotations.ApiModelProperty;
import lombok.Data;

import javax.persistence.Id;
import javax.persistence.Table;

/**
```

```
 * @ClassName CategoryEntity
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@ApiModel(value = "分类实体类")
@Data
@Table(name = "tb_category")
public class CategoryEntity {

    @Id
    @ApiModelProperty(value = "分类主键",example = "1")
    private Integer id;

    @ApiModelProperty(value = "分类名称")
    private String name;

    @ApiModelProperty(value = "父级分类",example = "1")
    private Integer parentId;

    @ApiModelProperty(value = "是否是父级节点",example = "1")
    private Integer isParent;

    @ApiModelProperty(value = "排序",example = "1")
    private Integer sort;
}
```

3.2.6 swagger2的配置

> com.baidu.shop.config.MrSwagger2Config

```
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import springfox.documentation.builders.ApiInfoBuilder;
import springfox.documentation.builders.PathSelectors;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.service.ApiInfo;
import springfox.documentation.service.Contact;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;
/**
 * @ClassName MrSwagger2Config
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@Configuration
@EnableSwagger2
public class MrSwagger2Config {

    @Bean
    public Docket createRestApi(){
        return new Docket(DocumentationType.SWAGGER_2)
                .apiInfo(this.apiInfo())
                .select()
                .apis(RequestHandlerSelectors.basePackage("com.baidu"))
                .paths(PathSelectors.any())
                .build();
    }

    private ApiInfo apiInfo(){
        return new ApiInfoBuilder()
                //标题
                .title("明瑞SWAGGER2标题")
                //条款地址
                .termsOfServiceUrl("http://www.baidu.com")
                //联系方式-->有String参数的方法但是已经过时，所以不推荐使用
                .contact(new
Contact("shenyaqi","baidu.com","shenyaqiii@163.com"))
                //版本
                .version("v1.0")
                //项目描述
                .description("描述")
                //创建API基本信息
                .build();
    }
}
```

3.2.7 定义商品分类接口

```
import com.baidu.shop.base.Result;
import com.baidu.shop.entity.CategoryEntity;
import io.swagger.annotations.Api;
import io.swagger.annotations.ApiOperation;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;

/**
 * @ClassName Category
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@Api(tags = "商品分类接口")
public interface CategoryService {

    @ApiOperation(value = "通过查询商品分类")
    @GetMapping(value = "category/list")
    Result<List<CategoryEntity>> getCategoryByPid(Integer pid);

}
```

3.3 service工程

3.3.1 mingrui-shop-service/pom.xml

> tkMapper :

> 文档：26-spring-boot-通用mapper.note
> 链接：http://note.youdao.com/noteshare?id=ab635b10d64562aa08c58804d2fe3629&sub=047B99D6647F40B2AFC66CB5BAAFB299

```xml
    <dependencies>

        <!-- SpringBoot-整合Web组件 -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <!-- springcloud feign组件 -->
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-starter-openfeign</artifactId>
        </dependency>

        <!--mysql数据库连接-->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <scope>runtime</scope><!--项目运行阶段使用-->
        </dependency>
        <!--通用mapper-->
        <dependency>
            <groupId>tk.mybatis</groupId>
            <artifactId>mapper-spring-boot-starter</artifactId>
            <version>2.1.5</version>
        </dependency>
        <!--分页工具-->
        <dependency>
            <groupId>com.github.pagehelper</groupId>
            <artifactId>pagehelper-spring-boot-starter</artifactId>
            <version>1.2.10</version>
        </dependency>

        <dependency>
            <groupId>com.baidu</groupId>
            <artifactId>mingrui-shop-service-api-xxx</artifactId>
            <version>1.0-SNAPSHOT</version>
        </dependency>


    </dependencies>
```

3.3.2 在mingrui-shop-service工程下新建mingrui-shop-service-xxx工程

3.3.3 application.yml

> hikari :
>
> 文档：9-spring-boot整合数据源以及mybatis.n...
> 链接：http://note.youdao.com/noteshare?id=e5b2180b5eaa2037ed90b1510ae9b546&sub=31EB54D010BB435C8A49450AE5A36493

```yaml
server:
  port: 8100

spring:
  application:
    name: xxx-server
  # 配置数据源
  datasource:
    # 数据源名称，任意
    name: mysql
    url: jdbc:mysql://ip:3306/mr-shop?
useSSL=true&nullNamePatternMatchesAll=true&serverTimezone=GMT%2B8&useUnicode=tru
e&characterEncoding=utf8
    # 数据库连接用户
    username: root
    # 数据库连接密码
    password: pwd
    # 驱动名称
    driver-class-name: com.mysql.cj.jdbc.Driver
    # boot2.0+使用hikari作为默认数据库连接池
    type: com.zaxxer.hikari.HikariDataSource
    hikari:
      # 是否自动提交事务 默认
      auto-commit: true
      # 允许的最小连接数
      minimum-idle: 5
      # 连接池内最大连接数
      maximum-pool-size: 10
      # 验证连接的sql语句
      connection-test-query: SELECT 1 FROM DUAL
      # 连接超时时间 默认30000 毫秒 如果小于250毫秒，则被重置回30秒
      connection-timeout: 30000
      # 验证超时时间默认5000毫秒 如果小于250毫秒，则会被重置回5秒
      validation-timeout: 5000
      # 设置连接在连接池中的存货时间 如果不等于0且小于30秒则会被重置回30分钟
      max-lifetime: 1800000

# 通用mapper
mapper:
  mappers: tk.mybatis.mapper.common.Mapper
  identity: MYSQL
#日志设置
logging:
  level:
    # 打印与我们程序相关的日志信息
    com.baidu.shop: debug
# eureka配置
eureka:
  client:
    service-url:
      defaultZone: http://localhost:8761/eureka/
```

3.3.4 新建包com.baidu

3.3.5 新建启动类

```
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.netflix.eureka.EnableEurekaClient;
import tk.mybatis.spring.annotation.MapperScan;

/**
 * @ClassName RunXXXApplication
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@SpringBootApplication
@EnableEurekaClient
@MapperScan("com.baidu.shop.mapper")
public class RunXXXApplication {
    public static void main(String[] args) {
        SpringApplication.run(RunXXXApplication.class);
    }
}
```

3.3.6 在com.baidu下新建包shop.mapper

3.3.7 创建mapper

```
import com.baidu.shop.entity.CategoryEntity;
import tk.mybatis.mapper.common.Mapper;

/**
 * @ClassName CategoryMapper
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
public interface CategoryMapper extends Mapper<CategoryEntity> {
}
```

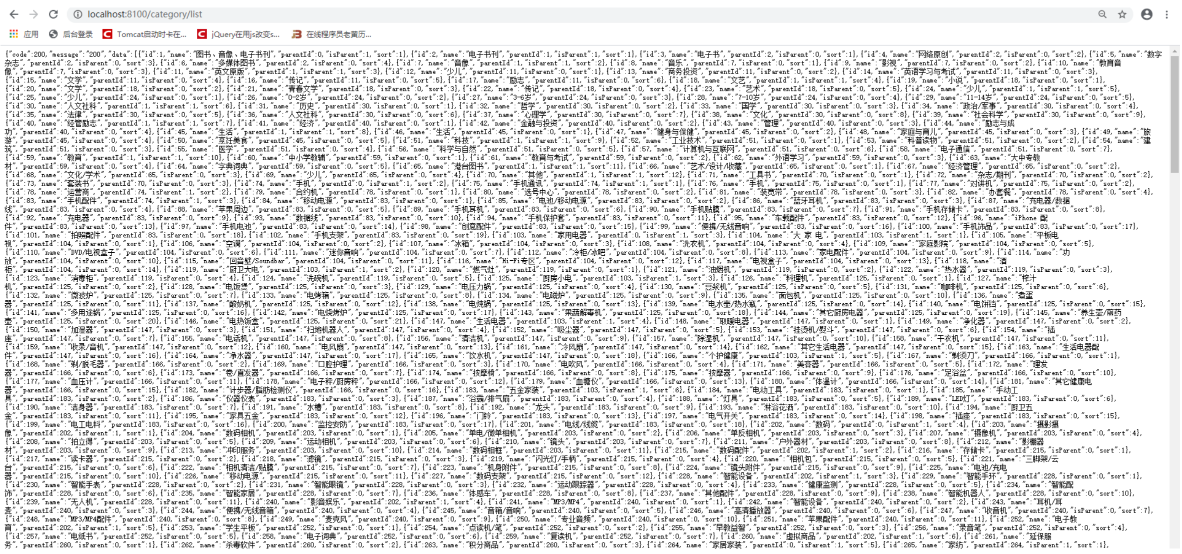
3.3.7 在com.baidu下新建包shop.service.impl

3.3.8 新建实现类

```
import com.baidu.shop.mapper.CategoryMapper;
import com.baidu.shop.base.BaseApiService;
import com.baidu.shop.base.Result;
import com.baidu.shop.entity.CategoryEntity;
import com.baidu.shop.service.CategoryService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RestController;

import java.util.List;

/**
 * @ClassName CategoryServiceImpl
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@RestController
public class CategoryServiceImpl extends BaseApiService implements
CategoryService {

    @Autowired
    private CategoryMapper categoryMapper;

    @Override
    public Result<List<CategoryEntity>> getCategoryByPid(Integer pid) {

        CategoryEntity categoryEntity = new CategoryEntity();

        categoryEntity.setParentId(pid);

        List<CategoryEntity> list = categoryMapper.select(categoryEntity);

        return this.setResultSuccess(list);
    }
}
```

3.3.9 浏览器输入

```
http://localhost:8100/category/list
```



```
{"code":200,"message":"200","data":[{"id":1,"name":"图书、音像、电子书刊","parentId":0,"isParent":1,"sort":1},{"id":2,"name":"电子书刊","parentId":1,"isParent":1,"sort":1},{"id":3,"name":"电子书","parentId":2,"isParent":0,"sort":1},{"id":4,"name":"网络原创","parentId":2,"isParent":0,"sort":2},{"id":5,"name":"数字
杂志","parentId":2,"isParent":0,"sort":3},{"id":6,"name":"多媒体图书","parentId":2,"isParent":0,"sort":4},{"id":7,"name":"音像","parentId":1,"isParent":1,"sort":2},{"id":8,"name":"音乐","parentId":7,"isParent":0,"sort":1},{"id":9,"name":"影视","parentId":7,"isParent":0,"sort":2},{"id":10,"name":"教育音
像","parentId":7,"isParent":0,"sort":3},{"id":11,"name":"英文原版","parentId":1,"isParent":1,"sort":3},{"id":12,"name":"少儿","parentId":11,"isParent":0,"sort":1},{"id":13,"name":"商务投资","parentId":11,"isParent":0,"sort":2},{"id":14,"name":"英语学习与考试","parentId":11,"isParent":0,"sort":3},
{"id":15,"name":"文学","parentId":11,"isParent":0,"sort":4},{"id":16,"name":"传记","parentId":11,"isParent":0,"sort":5},{"id":17,"name":"励志","parentId":11,"isParent":0,"sort":6},{"id":18,"name":"文艺","parentId":1,"isParent":1,"sort":4},{"id":19,"name":"小说","parentId":18,"isParent":0,"sort":1},
{"id":20,"name":"文学","parentId":18,"isParent":0,"sort":2},{"id":21,"name":"青春文学","parentId":18,"isParent":0,"sort":3},{"id":22,"name":"传记","parentId":18,"isParent":0,"sort":4},{"id":23,"name":"艺术","parentId":18,"isParent":0,"sort":5},{"id":24,"name":"少儿","parentId":1,"isParent":1,"sort":5},
{"id":25,"name":"少儿","parentId":24,"isParent":0,"sort":1},{"id":26,"name":"0-2岁","parentId":24,"isParent":0,"sort":2},{"id":27,"name":"3-6岁","parentId":24,"isParent":0,"sort":3},{"id":28,"name":"7-10岁","parentId":24,"isParent":0,"sort":4},{"id":29,"name":"11-14岁","parentId":24,"isParent":0,"sort":5},
{"id":30,"name":"人文社科","parentId":1,"isParent":1,"sort":6},{"id":31,"name":"历史","parentId":30,"isParent":0,"sort":1},{"id":32,"name":"哲学","parentId":30,"isParent":0,"sort":2},{"id":33,"name":"国学","parentId":30,"isParent":0,"sort":3},{"id":34,"name":"政治/军事","parentId":30,"isParent":0,"sort":4},
{"id":35,"name":"法律","parentId":30,"isParent":0,"sort":5},{"id":36,"name":"人文社科","parentId":30,"isParent":0,"sort":6},{"id":37,"name":"心理学","parentId":30,"isParent":0,"sort":7},{"id":38,"name":"文化","parentId":30,"isParent":0,"sort":8},{"id":39,"name":"社会科学","parentId":30,"isParent":0,"sort":9},
{"id":40,"name":"经管励志","parentId":1,"isParent":1,"sort":7},{"id":41,"name":"经济","parentId":40,"isParent":0,"sort":1},{"id":42,"name":"金融与投资","parentId":40,"isParent":0,"sort":2},{"id":43,"name":"管理","parentId":40,"isParent":0,"sort":3},{"id":44,"name":"励志与成
功","parentId":40,"isParent":0,"sort":4},{"id":45,"name":"生活","parentId":1,"isParent":1,"sort":8},{"id":46,"name":"生活","parentId":45,"isParent":0,"sort":1},{"id":47,"name":"健身与保健","parentId":45,"isParent":0,"sort":2},{"id":48,"name":"家庭与育儿","parentId":45,"isParent":0,"sort":3},{"id":49,"name":"旅
游","parentId":45,"isParent":0,"sort":4},{"id":50,"name":"烹饪美食","parentId":45,"isParent":0,"sort":5},{"id":51,"name":"科技","parentId":1,"isParent":1,"sort":9},{"id":52,"name":"工业技术","parentId":51,"isParent":0,"sort":1},{"id":53,"name":"科普读物","parentId":51,"isParent":0,"sort":2},{"id":54,"name":"建
筑","parentId":51,"isParent":0,"sort":3},{"id":55,"name":"医学","parentId":51,"isParent":0,"sort":4},{"id":56,"name":"科学与自然","parentId":51,"isParent":0,"sort":5},{"id":57,"name":"计算机与互联网","parentId":51,"isParent":0,"sort":6},{"id":58,"name":"电子通信","parentId":51,"isParent":0,"sort":7},
{"id":59,"name":"教育","parentId":1,"isParent":1,"sort":10},{"id":60,"name":"中小学教辅","parentId":59,"isParent":0,"sort":1},{"id":61,"name":"教育与考试","parentId":59,"isParent":0,"sort":2},{"id":62,"name":"外语学习","parentId":59,"isParent":0,"sort":3},{"id":63,"name":"大中专教
材","parentId":59,"isParent":0,"sort":4},{"id":64,"name":"字典词典","parentId":59,"isParent":0,"sort":5},{"id":65,"name":"港台图书","parentId":1,"isParent":1,"sort":11},{"id":66,"name":"艺术/设计/收藏","parentId":65,"isParent":0,"sort":1},{"id":67,"name":"经济管理","parentId":65,"isParent":0,"sort":2},
{"id":68,"name":"文化/学术","parentId":65,"isParent":0,"sort":3},{"id":69,"name":"少儿","parentId":65,"isParent":0,"sort":4},{"id":70,"name":"其他","parentId":1,"isParent":1,"sort":12},{"id":71,"name":"工具书","parentId":70,"isParent":0,"sort":1},{"id":72,"name":"杂志/期刊","parentId":70,"isParent":0,"sort":2},
{"id":73,"name":"套装书","parentId":70,"isParent":0,"sort":3},{"id":74,"name":"手机","parentId":0,"isParent":1,"sort":2},{"id":75,"name":"手机通讯","parentId":74,"isParent":1,"sort":1},{"id":76,"name":"手机","parentId":75,"isParent":0,"sort":1},{"id":77,"name":"对讲机","parentId":75,"isParent":0,"sort":2},
{"id":78,"name":"运营商","parentId":74,"isParent":1,"sort":2},{"id":79,"name":"合约机","parentId":78,"isParent":0,"sort":1},{"id":80,"name":"选号中心","parentId":78,"isParent":0,"sort":2},{"id":81,"name":"装宽带","parentId":78,"isParent":0,"sort":3},{"id":82,"name":"办套餐","parentId":78,"isParent":0,"sort":4},
{"id":83,"name":"手机配件","parentId":74,"isParent":1,"sort":3},{"id":84,"name":"移动电源","parentId":83,"isParent":0,"sort":1},{"id":85,"name":"电池/移动电源","parentId":83,"isParent":0,"sort":2},{"id":86,"name":"蓝牙耳机","parentId":83,"isParent":0,"sort":3},{"id":87,"name":"充电器/数据
线","parentId":83,"isParent":0,"sort":4},{"id":88,"name":"苹果周边","parentId":83,"isParent":0,"sort":5},{"id":89,"name":"手机耳机","parentId":83,"isParent":0,"sort":6},{"id":90,"name":"手机贴膜","parentId":83,"isParent":0,"sort":7},{"id":91,"name":"手机存储卡","parentId":83,"isParent":0,"sort":8},
{"id":92,"name":"充电器","parentId":83,"isParent":0,"sort":9},{"id":93,"name":"数据线","parentId":83,"isParent":0,"sort":10},{"id":94,"name":"手机保护套","parentId":83,"isParent":0,"sort":11},{"id":95,"name":"车载配件","parentId":83,"isParent":0,"sort":12},{"id":96,"name":"iPhone 配
件","parentId":83,"isParent":0,"sort":13},{"id":97,"name":"手机电池","parentId":83,"isParent":0,"sort":14},{"id":98,"name":"创意配件","parentId":83,"isParent":0,"sort":15},{"id":99,"name":"便携/无线音响","parentId":83,"isParent":0,"sort":16},{"id":100,"name":"手机饰品","parentId":83,"isParent":0,"sort":17},
{"id":101,"name":"拍照配件","parentId":83,"isParent":0,"sort":18},{"id":102,"name":"手机支架","parentId":83,"isParent":0,"sort":19},{"id":103,"name":"家用电器","parentId":0,"isParent":1,"sort":3},{"id":104,"name":"大 家 电","parentId":103,"isParent":1,"sort":1},{"id":105,"name":"平板电
视","parentId":104,"isParent":0,"sort":1},{"id":106,"name":"空调","parentId":104,"isParent":0,"sort":2},{"id":107,"name":"冰箱","parentId":104,"isParent":0,"sort":3},{"id":108,"name":"洗衣机","parentId":104,"isParent":0,"sort":4},{"id":109,"name":"家庭影院","parentId":104,"isParent":0,"sort":5},
{"id":110,"name":"DVD/电视盒子","parentId":104,"isParent":0,"sort":6},{"id":111,"name":"迷你音响","parentId":104,"isParent":0,"sort":7},{"id":112,"name":"冷柜/冰吧","parentId":104,"isParent":0,"sort":8},{"id":113,"name":"家电配件","parentId":104,"isParent":0,"sort":9},{"id":114,"name":"功
放","parentId":104,"isParent":0,"sort":10},{"id":115,"name":"回音壁/Soundbar","parentId":104,"isParent":0,"sort":11},{"id":116,"name":"Hi-Fi专区","parentId":104,"isParent":0,"sort":12},{"id":117,"name":"电视盒子","parentId":104,"isParent":0,"sort":13},{"id":118,"name":"酒
柜","parentId":104,"isParent":0,"sort":14},{"id":119,"name":"厨卫大电","parentId":103,"isParent":1,"sort":2},{"id":120,"name":"燃气灶","parentId":119,"isParent":0,"sort":1},{"id":121,"name":"油烟机","parentId":119,"isParent":0,"sort":2},{"id":122,"name":"热水器","parentId":119,"isParent":0,"sort":3},
{"id":123,"name":"消毒柜","parentId":119,"isParent":0,"sort":4},{"id":124,"name":"洗碗机","parentId":119,"isParent":0,"sort":5},{"id":125,"name":"厨房小电","parentId":103,"isParent":1,"sort":3},{"id":126,"name":"料理机","parentId":125,"isParent":0,"sort":1},{"id":127,"name":"榨汁
机","parentId":125,"isParent":0,"sort":2},{"id":128,"name":"电饭煲","parentId":125,"isParent":0,"sort":3},{"id":129,"name":"电压力锅","parentId":125,"isParent":0,"sort":4},{"id":130,"name":"豆浆机","parentId":125,"isParent":0,"sort":5},{"id":131,"name":"咖啡机","parentId":125,"isParent":0,"sort":6},
{"id":132,"name":"微波炉","parentId":125,"isParent":0,"sort":7},{"id":133,"name":"电烤箱","parentId":125,"isParent":0,"sort":8},{"id":134,"name":"电磁炉","parentId":125,"isParent":0,"sort":9},{"id":135,"name":"面包机","parentId":125,"isParent":0,"sort":10},{"id":136,"name":"煮蛋
器","parentId":125,"isParent":0,"sort":11},{"id":137,"name":"酸奶机","parentId":125,"isParent":0,"sort":12},{"id":138,"name":"电炖锅","parentId":125,"isParent":0,"sort":13},{"id":139,"name":"电水壶/热水瓶","parentId":125,"isParent":0,"sort":14},{"id":140,"name":"电饼铛","parentId":125,"isParent":0,"sort":15},
{"id":141,"name":"多用途锅","parentId":125,"isParent":0,"sort":16},{"id":142,"name":"电烧烤炉","parentId":125,"isParent":0,"sort":17},{"id":143,"name":"果蔬解毒机","parentId":125,"isParent":0,"sort":18},{"id":144,"name":"其它厨房电器","parentId":125,"isParent":0,"sort":19},{"id":145,"name":"养生壶/煎药
壶","parentId":125,"isParent":0,"sort":20},{"id":146,"name":"电热饭盒","parentId":125,"isParent":0,"sort":21},{"id":147,"name":"生活电器","parentId":103,"isParent":1,"sort":4},{"id":148,"name":"取暖电器","parentId":147,"isParent":0,"sort":1},{"id":149,"name":"净化器","parentId":147,"isParent":0,"sort":2},
{"id":150,"name":"加湿器","parentId":147,"isParent":0,"sort":3},{"id":151,"name":"扫地机器人","parentId":147,"isParent":0,"sort":4},{"id":152,"name":"吸尘器","parentId":147,"isParent":0,"sort":5},{"id":153,"name":"挂烫机/熨斗","parentId":147,"isParent":0,"sort":6},{"id":154,"name":"插
座","parentId":147,"isParent":0,"sort":7},{"id":155,"name":"电话机","parentId":147,"isParent":0,"sort":8},{"id":156,"name":"清洁机","parentId":147,"isParent":0,"sort":9},{"id":157,"name":"除湿机","parentId":147,"isParent":0,"sort":10},{"id":158,"name":"干衣机","parentId":147,"isParent":0,"sort":11},
{"id":159,"name":"收录/音机","parentId":147,"isParent":0,"sort":12},{"id":160,"name":"电风扇","parentId":147,"isParent":0,"sort":13},{"id":161,"name":"冷风扇","parentId":147,"isParent":0,"sort":14},{"id":162,"name":"其它生活电器","parentId":147,"isParent":0,"sort":15},{"id":163,"name":"生活电器配
件","parentId":147,"isParent":0,"sort":16},{"id":164,"name":"净水器","parentId":147,"isParent":0,"sort":17},{"id":165,"name":"饮水机","parentId":147,"isParent":0,"sort":18},{"id":166,"name":"个护健康","parentId":103,"isParent":1,"sort":5},{"id":167,"name":"剃须刀","parentId":166,"isParent":0,"sort":1},
{"id":168,"name":"剃/脱毛器","parentId":166,"isParent":0,"sort":2},{"id":169,"name":"口腔护理","parentId":166,"isParent":0,"sort":3},{"id":170,"name":"电吹风","parentId":166,"isParent":0,"sort":4},{"id":171,"name":"美容器","parentId":166,"isParent":0,"sort":5},{"id":172,"name":"理发
器","parentId":166,"isParent":0,"sort":6},{"id":173,"name":"卷/直发器","parentId":166,"isParent":0,"sort":7},{"id":174,"name":"按摩椅","parentId":166,"isParent":0,"sort":8},{"id":175,"name":"按摩器","parentId":166,"isParent":0,"sort":9},{"id":176,"name":"足浴盆","parentId":166,"isParent":0,"sort":10},
{"id":177,"name":"血压计","parentId":166,"isParent":0,"sort":11},{"id":178,"name":"电子秤/厨房秤","parentId":166,"isParent":0,"sort":12},{"id":179,"name":"血糖仪","parentId":166,"isParent":0,"sort":13},{"id":180,"name":"体温计","parentId":166,"isParent":0,"sort":14},{"id":181,"name":"其它健康电
器","parentId":166,"isParent":0,"sort":15},{"id":182,"name":"计步器/脂肪检测仪","parentId":166,"isParent":0,"sort":16},{"id":183,"name":"五金家装","parentId":103,"isParent":1,"sort":6},{"id":184,"name":"电动工具","parentId":183,"isParent":0,"sort":1},{"id":185,"name":"手动工
具","parentId":183,"isParent":0,"sort":2},{"id":186,"name":"仪器仪表","parentId":183,"isParent":0,"sort":3},{"id":187,"name":"浴霸/排气扇","parentId":183,"isParent":0,"sort":4},{"id":188,"name":"灯具","parentId":183,"isParent":0,"sort":5},{"id":189,"name":"LED灯","parentId":183,"isParent":0,"sort":6},
{"id":190,"name":"洁身器","parentId":183,"isParent":0,"sort":7},{"id":191,"name":"水槽","parentId":183,"isParent":0,"sort":8},{"id":192,"name":"龙头","parentId":183,"isParent":0,"sort":9},{"id":193,"name":"淋浴花洒","parentId":183,"isParent":0,"sort":10},{"id":194,"name":"厨卫五
金","parentId":183,"isParent":0,"sort":11},{"id":195,"name":"家具五金","parentId":183,"isParent":0,"sort":12},{"id":196,"name":"门铃","parentId":183,"isParent":0,"sort":13},{"id":197,"name":"电气开关","parentId":183,"isParent":0,"sort":14},{"id":198,"name":"插座","parentId":183,"isParent":0,"sort":15},
{"id":199,"name":"电工电料","parentId":183,"isParent":0,"sort":16},{"id":200,"name":"监控安防","parentId":183,"isParent":0,"sort":17},{"id":201,"name":"电线/线缆","parentId":183,"isParent":0,"sort":18},{"id":202,"name":"数码","parentId":0,"isParent":1,"sort":4},{"id":203,"name":"摄影摄
像","parentId":202,"isParent":1,"sort":1},{"id":204,"name":"数码相机","parentId":203,"isParent":0,"sort":1},{"id":205,"name":"单电/微单相机","parentId":203,"isParent":0,"sort":2},{"id":206,"name":"单反相机","parentId":203,"isParent":0,"sort":3},{"id":207,"name":"摄像机","parentId":203,"isParent":0,"sort":4},
{"id":208,"name":"拍立得","parentId":203,"isParent":0,"sort":5},{"id":209,"name":"运动相机","parentId":203,"isParent":0,"sort":6},{"id":210,"name":"镜头","parentId":203,"isParent":0,"sort":7},{"id":211,"name":"户外器材","parentId":203,"isParent":0,"sort":8},{"id":212,"name":"影棚器
材","parentId":203,"isParent":0,"sort":9},{"id":213,"name":"冲印服务","parentId":203,"isParent":0,"sort":10},{"id":214,"name":"数码相框","parentId":203,"isParent":0,"sort":11},{"id":215,"name":"数码配件","parentId":202,"isParent":1,"sort":2},{"id":216,"name":"存储卡","parentId":215,"isParent":0,"sort":1},
{"id":217,"name":"读卡器","parentId":215,"isParent":0,"sort":2},{"id":218,"name":"滤镜","parentId":215,"isParent":0,"sort":3},{"id":219,"name":"闪光灯/手柄","parentId":215,"isParent":0,"sort":4},{"id":220,"name":"相机包","parentId":215,"isParent":0,"sort":5},{"id":221,"name":"三脚架/云
台","parentId":215,"isParent":0,"sort":6},{"id":222,"name":"相机清洁/贴膜","parentId":215,"isParent":0,"sort":7},{"id":223,"name":"机身附件","parentId":215,"isParent":0,"sort":8},{"id":224,"name":"镜头附件","parentId":215,"isParent":0,"sort":9},{"id":225,"name":"电池/充电
器","parentId":215,"isParent":0,"sort":10},{"id":226,"name":"移动电源","parentId":215,"isParent":0,"sort":11},{"id":227,"name":"数码支架","parentId":215,"isParent":0,"sort":12},{"id":228,"name":"智能设备","parentId":202,"isParent":1,"sort":3},{"id":229,"name":"智能手环","parentId":228,"isParent":0,"sort":1},
{"id":230,"name":"智能手表","parentId":228,"isParent":0,"sort":2},{"id":231,"name":"智能眼镜","parentId":228,"isParent":0,"sort":3},{"id":232,"name":"运动跟踪器","parentId":228,"isParent":0,"sort":4},{"id":233,"name":"健康监测","parentId":228,"isParent":0,"sort":5},{"id":234,"name":"智能配
饰","parentId":228,"isParent":0,"sort":6},{"id":235,"name":"智能家居","parentId":228,"isParent":0,"sort":7},{"id":236,"name":"体感车","parentId":228,"isParent":0,"sort":8},{"id":237,"name":"其他配件","parentId":228,"isParent":0,"sort":9},{"id":238,"name":"智能机器人","parentId":228,"isParent":0,"sort":10},
{"id":239,"name":"无人机","parentId":228,"isParent":0,"sort":11},{"id":240,"name":"影音娱乐","parentId":202,"isParent":1,"sort":4},{"id":241,"name":"MP3/MP4","parentId":240,"isParent":0,"sort":1},{"id":242,"name":"智能设备","parentId":240,"isParent":0,"sort":2},{"id":243,"name":"耳机/耳
麦","parentId":240,"isParent":0,"sort":3},{"id":244,"name":"便携/无线音箱","parentId":240,"isParent":0,"sort":4},{"id":245,"name":"音箱/音响","parentId":240,"isParent":0,"sort":5},{"id":246,"name":"高清播放器","parentId":240,"isParent":0,"sort":6},{"id":247,"name":"收音机","parentId":240,"isParent":0,"sort":7},
{"id":248,"name":"MP3/MP4配件","parentId":240,"isParent":0,"sort":8},{"id":249,"name":"麦克风","parentId":240,"isParent":0,"sort":9},{"id":250,"name":"专业音频","parentId":240,"isParent":0,"sort":10},{"id":251,"name":"苹果配件","parentId":240,"isParent":0,"sort":11},{"id":252,"name":"电子教
育","parentId":202,"isParent":1,"sort":5},{"id":253,"name":"学生平板","parentId":252,"isParent":0,"sort":1},{"id":254,"name":"点读机/笔","parentId":252,"isParent":0,"sort":2},{"id":255,"name":"早教益智","parentId":252,"isParent":0,"sort":3},{"id":256,"name":"录音笔","parentId":252,"isParent":0,"sort":4},
{"id":257,"name":"电纸书","parentId":252,"isParent":0,"sort":5},{"id":258,"name":"电子词典","parentId":252,"isParent":0,"sort":6},{"id":259,"name":"复读机","parentId":252,"isParent":0,"sort":7},{"id":260,"name":"虚拟商品","parentId":202,"isParent":1,"sort":6},{"id":261,"name":"延保服
务","parentId":260,"isParent":0,"sort":1},{"id":262,"name":"杀毒软件","parentId":260,"isParent":0,"sort":2},{"id":263,"name":"积分商品","parentId":260,"isParent":0,"sort":3},{"id":264,"name":"家居家装","parentId":0,"isParent":1,"sort":5},{"id":265,"name":"家纺","parentId":264,"isParent":1,"sort":1},
```

3.4 网关服务

3.4.1 在mingrui-shop-basic项目下新建mingrui-shop-basic-zuul-server

3.4.2 pom.xml

```xml
    <dependencies>

        <!-- zuul -->
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-starter-netflix-zuul</artifactId>
        </dependency>

    </dependencies>
```

3.4.3 application.yml

```yaml
server:
  port: 8088

spring:
  application:
    name: eureka-zuul

zuul:
  # 声明路由
  routes:
    # 路由名称
    api-xxx:
      # 声明将所有以/api-ribbon/的请求都转发到eureka-ribbon的服务中
      path: /api-xxx/**
      serviceId: xxx-server
  # 启用重试
  retryable: true
#配置负载
ribbon:
  ConnectTimeout: 250 # 连接超时时间(ms)
  ReadTimeout: 2000 # 通信超时时间(ms)
  OkToRetryOnAllOperations: true # 是否对所有操作重试
  MaxAutoRetriesNextServer: 2 # 同一服务不同实例的重试次数
  MaxAutoRetries: 1 # 同一实例的重试次数

hystrix:
  command:
    default:
      execution:
        isolation:
          thread:
            timeoutInMilliseconds: 10000 # 熔断超时时长：6000ms

eureka:
  client:
    service-url:
      defaultZone: http://localhost:8761/eureka/
```

3.4.4 新建包com.baidu

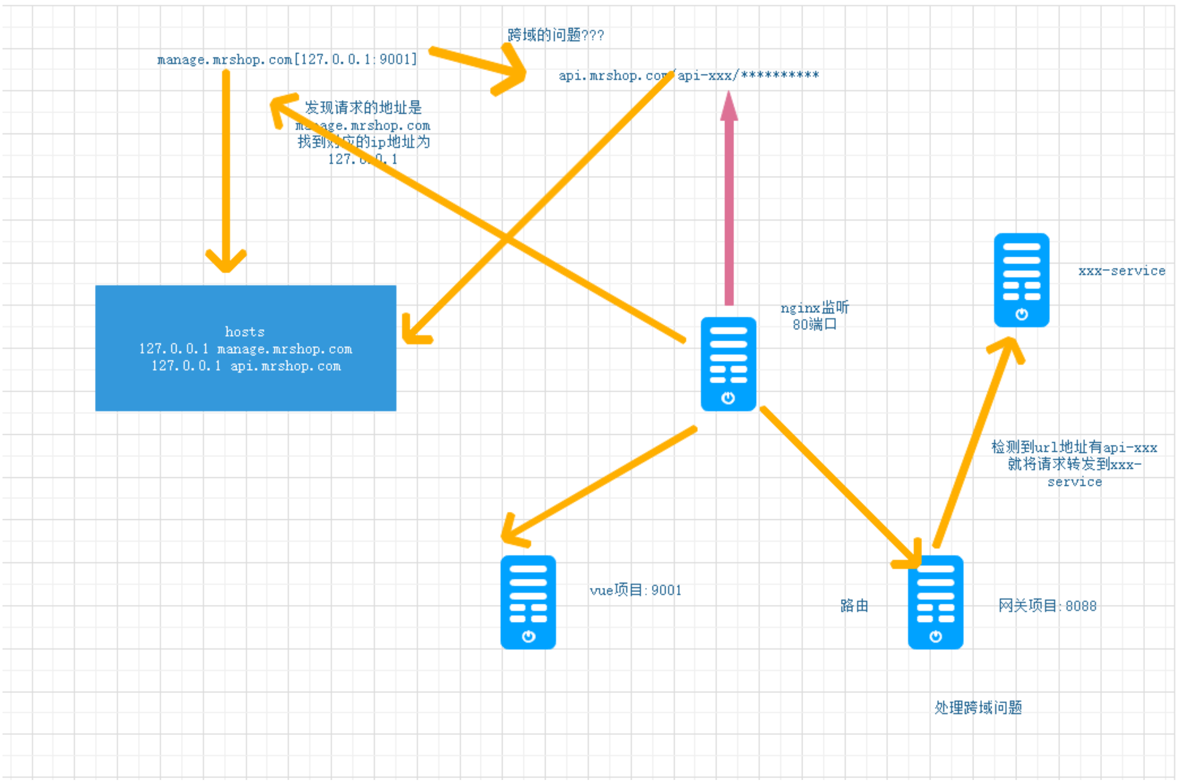
3.4.5 新建启动类

```
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.netflix.eureka.EnableEurekaClient;
import org.springframework.cloud.netflix.zuul.EnableZuulProxy;

/**
 * @ClassName RunZuulServerApplication
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/14
 * @Version V1.0
 **/
@SpringBootApplication
@EnableZuulProxy
@EnableEurekaClient
public class RunZuulServerApplication {

    public static void main(String[] args) {
        SpringApplication.run(RunZuulServerApplication.class);
    }
}
```

3.4.6 在com.baidu下新建global.GlobalCorsConfig

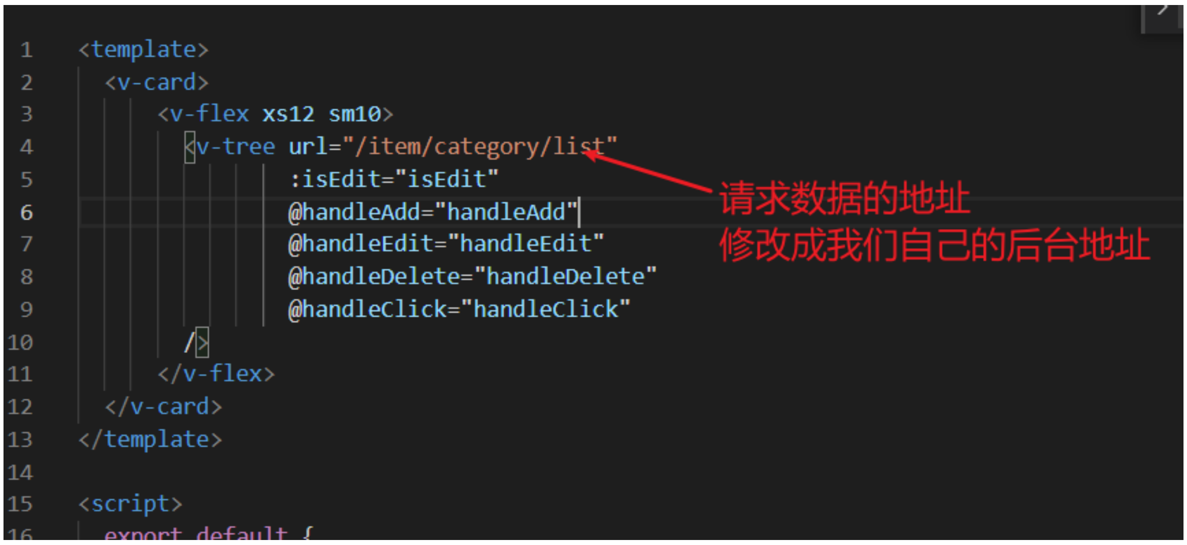
```
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.cors.CorsConfiguration;
import org.springframework.web.cors.UrlBasedCorsConfigurationSource;
import org.springframework.web.filter.CorsFilter;

/**
 * @ClassName GlobalCorsConfig
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/28
 * @Version V1.0
 **/
@Configuration
public class GlobalCorsConfig {

    @Bean
    public CorsFilter corsFilter() {
        final UrlBasedCorsConfigurationSource source = new
UrlBasedCorsConfigurationSource();
        final CorsConfiguration config = new CorsConfiguration();
        config.setAllowCredentials(true); // 允许cookies跨域
        config.addAllowedOrigin("*");// 允许向该服务器提交请求的URI，*表示全部允许。。
这里尽量限制来源域，比如http://xxxx:8080 ,以降低安全风险。。
        config.addAllowedHeader("*");// 允许访问的头信息,*表示全部
        config.setMaxAge(18000L);// 预检请求的缓存时间（秒），即在这个时间段里，对于相同
的跨域请求不会再预检了
        config.addAllowedMethod("*");// 允许提交请求的方法，*表示全部允许，也可以单独设
置GET、PUT等
        config.addAllowedMethod("HEAD");
        config.addAllowedMethod("GET");// 允许Get的请求方法
        config.addAllowedMethod("PUT");
```

```java
        config.addAllowedMethod("POST");
        config.addAllowedMethod("DELETE");
        config.addAllowedMethod("PATCH");
        source.registerCorsConfiguration("/**", config);
        //3.返回新的CorsFilter.
        return new CorsFilter(source);
    }
}
```

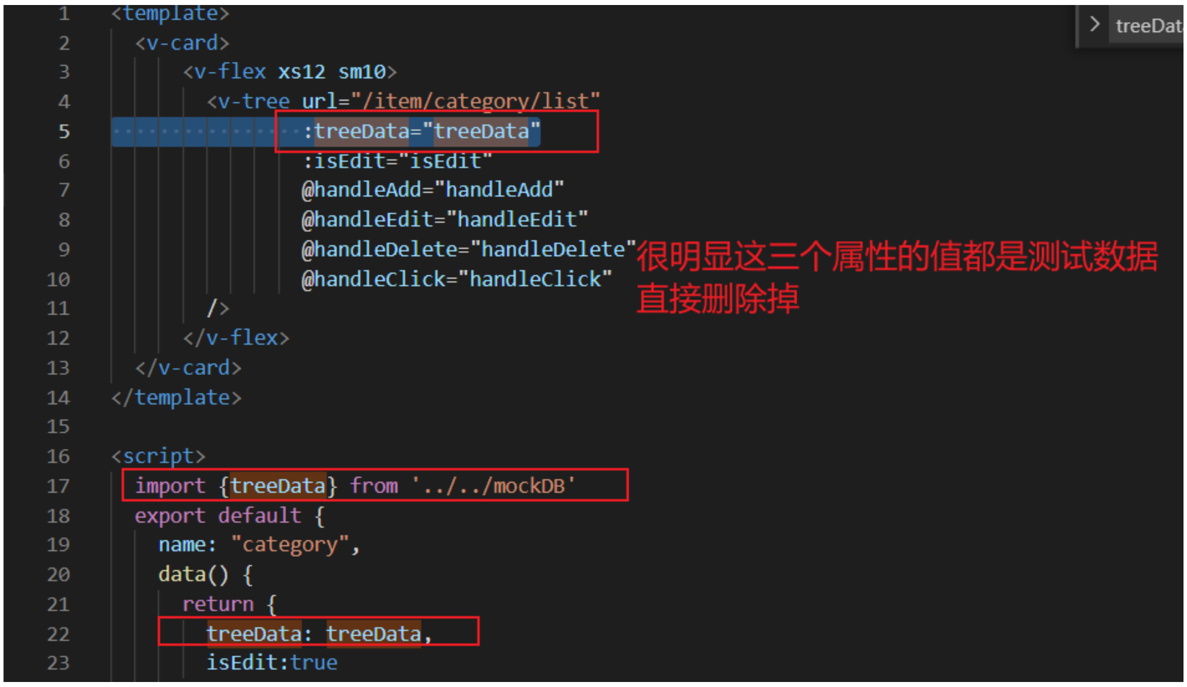

manage.mrshop.com[127.0.0.1:9001]
跨域的问题???
api.mrshop.com/api-xxx/**********
发现请求的地址是 manage.mrshop.com 找到对应的ip地址为 127.0.0.1
hosts
127.0.0.1 manage.mrshop.com
127.0.0.1 api.mrshop.com
nginx监听 80端口
xxx-service
检测到url地址有api-xxx 就将请求转发到xxx-service
vue项目:9001
路由
网关项目:8088
处理跨域问题

3.5 vue项目

3.5.1 Category.vue



```html
<template>
  <v-card>
      <v-flex xs12 sm10>
        <v-tree url="/item/category/list"
                :isEdit="isEdit"
                @handleAdd="handleAdd"
                @handleEdit="handleEdit"
                @handleDelete="handleDelete"
                @handleClick="handleClick"
        />
      </v-flex>
  </v-card>
</template>

<script>
  export default {
```



```
1   <template>
2     <v-card>
3       <v-flex xs12 sm10>
4         <v-tree url="/item/category/list"
5                 :treeData="treeData"
6                 :isEdit="isEdit"
7                 @handleAdd="handleAdd"
8                 @handleEdit="handleEdit"
9                 @handleDelete="handleDelete"
10                @handleClick="handleClick"
11        />
12      </v-flex>
13    </v-card>
14  </template>
15
16  <script>
17    import {treeData} from '../../mockDB'
18    export default {
19      name: "category",
20      data() {
21        return {
22          treeData: treeData,
23          isEdit:true
```

很明显这三个属性的值都是测试数据直接删除掉

删除line:5和line:22

```
<template>
  <v-card>
      <v-flex xs12 sm10>
        <v-tree url="/category/list"
                :isEdit="isEdit"
                @handleAdd="handleAdd"
                @handleEdit="handleEdit"
                @handleDelete="handleDelete"
                @handleClick="handleClick"
        />
      </v-flex>
  </v-card>
</template>

<script>
  //import {treeData} from '../../mockDB'
  export default {
    name: "category",
    data() {
      return {
        isEdit:true
      }
    },
    methods: {
      handleAdd(node) {
        console.log("add .... ");
        console.log(node);
      },
      handleEdit(id, name) {
        console.log("edit... id: " + id + ", name: " + name)
      },
      handleDelete(id) {
        console.log("delete ... " + id)
      },
      handleClick(node) {
        console.log(node)
      }
    }
  };
</script>

<style scoped>

</style>
```

3.5.2 Tree.vue

getData方法重新制定resp回调

```
<template>
  <v-list class="pt-0 pb-0" dense>
    <TreeItem
      class="item" :model="model" v-for="(model, index) in db" :key="index"
      :url="url"
      :isEdit="isEdit"
      :nodes="nodes"
      @handleAdd="handleAdd"
      @handleEdit="handleEdit"
      @handleDelete="handleDelete"
      @handleClick="handleClick"
    />
  </v-list>
</template>

<script>
  import TreeItem from './TreeItem';

  export default {
    name: "vTree",
    props: {
      url: String,
      isEdit: {
        type: Boolean,
        default: false
      },
      treeData:{
        type:Array
      }
    },
    data() {
      return {
        db: [],
        nodes:{
          opened:null,
          selected:{isSelected:false}
        }
      }
    },
    components: {
      TreeItem
    },
    created() {
      if(this.treeData && this.treeData.length > 0){
        this.db = this.treeData;
        return;
      }
      this.getData();
    },
    methods: {
      getData() {
        //加载一级菜单
        this.$http.get(this.url, {params: {pid: 0}}).then(resp => {
          console.log(resp)
          this.db = resp.data.data;
          this.db.forEach(n => n['path'] = [n.name])
        })
      },
      handleAdd(node) {
        this.$emit("handleAdd", this.copyNodeInfo(node));
      },
      handleEdit(id, name) {
        this.$emit("handleEdit", id, name)
      },
      handleDelete(id) {
        this.deleteById(id, this.db);
        this.$emit("handleDelete", id);
      },
      handleClick(node){
        this.$emit("handleClick", this.copyNodeInfo(node))
      },
      // 根据id删除
      deleteById(id, arr) {
        let src = arr || this.db;
        for (let i in src) {
          let d = src[i];
          if (d.id === id) {
            src.splice(i, 1)
            return;
          }
          if (d.children && d.children.length > 0) {
            this.deleteById(id, d.children)
          }
        }
      },
      copyNodeInfo(node){
        const o = {};
        for(let i in node){
          o[i] = node[i];
        }
        return o;
      }
    },
    watch: {}
  }
</script>

<style scoped>
  .item {
    cursor: pointer;
  }
```

```
</style>
```

3.5.3 TreeItem.vue

line:121 修改回调

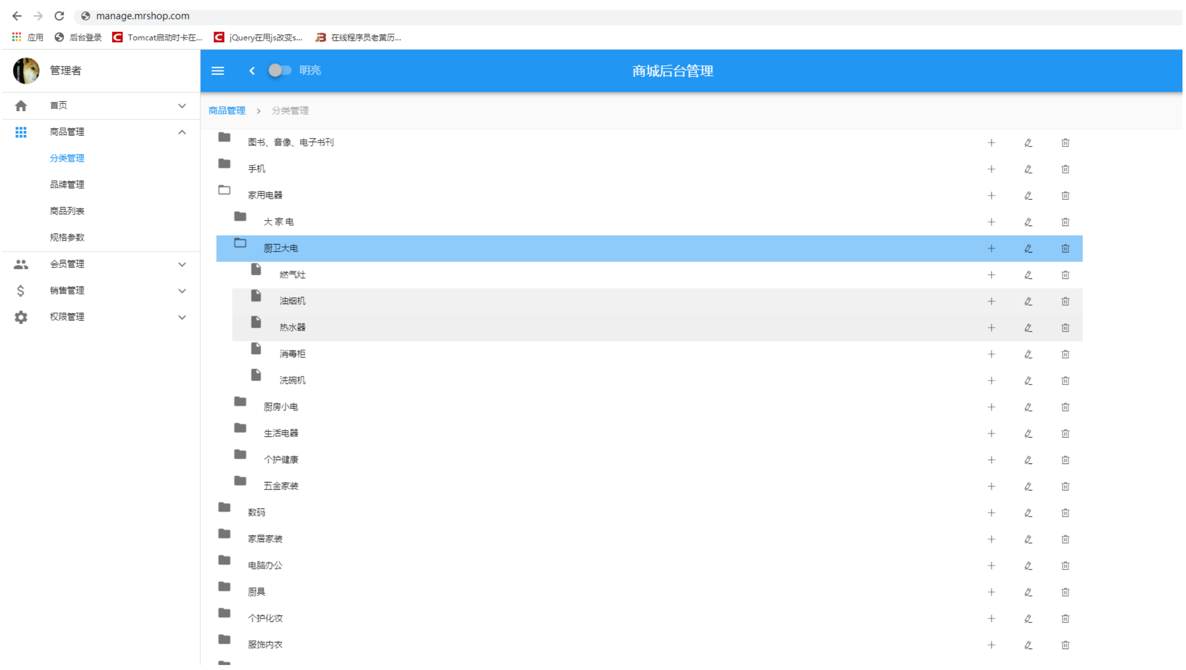
```
<template>
  <div>
    <v-list-tile
      @click="toggle" class="level1 pt-0 pb-0 mt-0 mb-0" :class="
{'selected':isSelected}">
      <v-list-tile-avatar>
        <v-icon v-if="model.isParent">{{open ? 'folder_open' : 'folder'}}</v-
icon>
        <v-icon v-if="!model.isParent">insert_drive_file</v-icon>
      </v-list-tile-avatar>
      <v-list-tile-content>
        <v-list-tile-title v-show="!beginEdit">
          <span >{{model.name}}</span>
        </v-list-tile-title>
        <input v-show="beginEdit" @click.stop="" :ref="model.id" v-
model="model.name"
               @blur="afterEdit" @keydown.enter="afterEdit"/>
      </v-list-tile-content>
      <v-list-tile-action v-if="isEdit">
        <v-btn icon @mouseover="c1='primary'" @mouseout="c1=''" :color="c1"
@click.stop="addChild">
          <i class="el-icon-plus"/>
        </v-btn>
      </v-list-tile-action>
      <v-list-tile-action v-if="isEdit">
        <v-btn icon @mouseover="c2='success'" @mouseout="c2=''" :color="c2"
@click.stop="editChild">
          <i class="el-icon-edit"/>
        </v-btn>
      </v-list-tile-action>
      <v-list-tile-action v-if="isEdit">
        <v-btn icon @mouseover="c3='error'" @mouseout="c3=''" :color="c3"
@click.stop="deleteChild">
          <i class="el-icon-delete"/>
        </v-btn>
      </v-list-tile-action>
    </v-list-tile>

    <v-list v-if="isFolder" v-show="open" class="ml-4 pt-0 pb-0" dense
transition="scale-transition">
      <tree-item
        class="item"
        v-for="(model, index) in model.children"
        :key="index"
        :model="model"
        :url="url"
        :isEdit="isEdit"
        :nodes="nodes"
        :parentState="open"
        @handleAdd="handleAdd"
        @handleEdit="handleEdit"
        @handleDelete="handleDelete"
        @handleClick="handleClick"
      >
      </tree-item>
    </v-list>
  </div>
</template>

<script>
  import Vue from 'vue'

  export default {
    name: "tree-item",
    props: {
      model: Object,
      url: String,
      isEdit: {
        type: Boolean,
        default: false
      },
      nodes: Object,
      parentState:Boolean
    },
    created() {
    },
    data: function () {
      return {
        c1: '',
        c2: '',
        c3: '',
        isSelected: false,
        open:false,
        beginEdit:false
      }
    },
    watch:{
      parentState(val){
        if(!val){
          this.open = val;
        }
      }
    },
    computed: {
      isFolder: function () {
        return this.model.children &&
          this.model.children.length > 0
      }
    },
    methods: {
      toggle: function () {
        // 将其它被选中项取消选中
        this.nodes.selected.isSelected = false;
        // 当前项被选中
        this.isSelected = true;
        // 保存当前选中项
        this.nodes.selected = this
        // 客户自己的点击事件回调
        this.handleClick(this.model);

        // 判断是否为顶级节点，顶级节点需要记录和替换
        if(this.model.parentId == 0){
          // 判断打开节点是否是自己
          if(this.nodes.opened && this != this.nodes.opened){
            // 不是，则关闭原来的节点
            this.nodes.opened.open = false;
          }
          // 将自己记录为打开的节点
          this.nodes.opened = this;
        }
        // 切换开闭状态
        this.open = !this.open;
        // 如果已经是叶子节点,或者自己是关闭的，或者自己已经有儿子了，结束
        if (!this.model.isParent || this.isFolder || !this.open) {
          return;
        }
        // 展开后查询子节点
        //点击父节点信息回调
        this.$http.get(this.url, {params: {pid: this.model.id}})
          .then(resp => {
          Vue.set(this.model, 'children', resp.data.data);
          // 封装当前节点的路径
          this.model.children.forEach(n => {
            n['path'] = [];
            this.model.path.forEach(p => n['path'].push(p));
            n['path'].push(n.name);
          });
        }).catch( e => {
          console.log(e);
        });
      },
      addChild: function () {
        let child = {
          id: 0,
          name: '新的节点',
          parentId: this.model.id,
          isParent: false,
          sort:this.model.children? this.model.children.length + 1:1
        }
        if (!this.model.isParent) {
          Vue.set(this.model, 'children', [child]);
          this.model.isParent = true;
          this.open = true;
          this.handleAdd(child);
        } else {
          if (!this.isFolder) {
            this.$http.get(this.url, {params: {pid: this.model.id}}).then(resp
=> {
              Vue.set(this.model, 'children', resp.data.data);
              this.model.children.push(child);
              this.open = true;
              this.handleAdd(child);
            });
          } else {
            this.model.children.push(child);
            this.open = true;
            this.handleAdd(child);
          }
        }
      },
      deleteChild: function () {
        this.$message.confirm('此操作将永久删除数据，是否继续?', '提示', {
          confirmButtonText: '确定删除',
          cancelButtonText: '取消',
          type: 'warning'
        }).then(() => {
          this.handleDelete(this.model.id);
        }).catch(()=>{
          this.$message.info('已取消删除');
        })

      },
      editChild() {
        this.beginEdit = true;
        this.$nextTick(() => this.$refs[this.model.id].focus());
      },
      afterEdit() {
        if (this.model.beginEdit) {
          this.beginEdit = false;
          this.handleEdit(this.model.id, this.model.name);
        }
      },
      handleAdd(node) {
        this.$emit("handleAdd", node);
      },
      handleDelete(id) {
        this.$emit("handleDelete", id);
      },
      handleEdit(id, name) {
        this.$emit("handleEdit", id, name)
      },
      handleClick(node) {
        this.$emit("handleClick", node);
      }
    }
  }
</script>

<style scoped>
  .level1 {
    height: 40px;
  }

  .selected {
    background-color: rgba(105,184,249,0.75);
  }
</style>
```

依次启动eureka-server,xxx-service,zuul-server,vue项目

保证自己的hosts,nginx没有问题

浏览器输入

```
http://manage.mrshop.com/
```



4 商品分类管理(删除)

4.1 vue项目

4.1.1 TreeItem.vue : line 164

```
        deleteChild: function () {

          if(this.model.isParent == 1){
              this.$message.warning('当前节点为父节点');
              return ;
            }

          this.$message.confirm('此操作将永久删除数据，是否继续?', '提示', {
            confirmButtonText: '确定删除',
            cancelButtonText: '取消',
            type: 'warning'
          }).then(() => {
            this.handleDelete(this.model.id);
          }).catch(()=>{
            this.$message.info('已取消删除');
          })

        }
```

4.1.2 Category.vue

```
      handleDelete(id) {
        console.log("delete ... " + id);
        this.$http.delete('/category/del?id=' + id).then(response => {
          this.$message.success('删除成功');
          //重新加载树形组件
          this.key = new Date().getTime();//避免key重复
        }).catch(error => console.log(error))
      },
```

4.1.3 注意需要给v-tree组件绑定一个key的属性,默认值为0

```
     <v-tree :key="key" url="/category/list">
    data() {
      return {
        key:0
      }
    }
```

4.1.4 删除完成后给key重新设置一个值可实现组件刷新

4.2 mingrui-shop-service-api-xxx

4.2.1 CategoryService

```
    @ApiOperation(value = "通过id删除分类")
    @DeleteMapping(value = "category/del")
    public Result<JsonObject> delCategory(Integer id);
```

4.3 mingrui-shop-common-core

4.3.1 新建ObjectUtil(让代码更优雅)

```
/**
 * @ClassName 对象工具类
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
public class ObjectUtil {

    public static Boolean isNull(Object obj){

        return null == obj;
    }
}
```

4.4 common-core

4.4.1 HTTPStatus

```
public static final int OPERATION_ERROR = 5001;//操作失败
```

4.5 mingrui-shop-service-xxx

4.5.1 实现类中要做的事

- 通过当前id查询分类信息
- 判断是否有数据(安全)
- 判断当前节点是否是父级节点(安全)
- 判断当前节点的父节点下 除了当前节点是否还有别的节点(业务)
  - 没有:将当前节点的父节点isParent的值修改为0
- 通过id删除数据

4.5.2 代码实现

```
     @Override
    public Result<JSONObject> deleteCategory(Integer id) {
        //验证传入的id是否有效,并且查询出来的数据对接下来的程序有用
        CategoryEntity categoryEntity = categoryMapper.selectByPrimaryKey(id);
        if (categoryEntity == null) {
            return this.setResultError("当前id不存在");
        }
        //判断当前节点是否为父节点
        if(categoryEntity.getIsParent() == 1){
            return this.setResultError("当前节点为父节点,不能删除");
        }

        //构建条件查询 通过当前被删除节点的parentid查询数据
        Example example = new Example(CategoryEntity.class);

example.createCriteria().andEqualTo("parentId",categoryEntity.getParentId());
        List<CategoryEntity> list = categoryMapper.selectByExample(example);
        //如果查询出来的数据只有一条
        if(list.size() == 1){//将父节点的isParent状态改为0

            CategoryEntity parentCateEntity = new CategoryEntity();
            parentCateEntity.setId(categoryEntity.getParentId());
            parentCateEntity.setIsParent(0);
            categoryMapper.updateByPrimaryKeySelective(parentCateEntity);
        }

        categoryMapper.deleteByPrimaryKey(id);//执行删除

        return this.setResultSuccess();
    }
```

5 商品分类管理(修改)

5.1 vue项目

5.2.1 将TreeItem.vue的afterEdit方法this.model.beginEdit改为this.beginEdit

```
      afterEdit() {
        if (this.beginEdit) {//bug,(this.model.beginEdit)哎毕竟用人家的东西.....
          this.beginEdit = false;
          this.handleEdit(this.model.id, this.model.name);
        }
      },
```

5.1.2 Category.vue

```
      handleEdit(id, name) {
        console.log("edit... id: " + id + ", name: " + name)
        //axios提交
        this.$http.put('/category/edit',{
          id:id,
          name:name
        }).then(resp => {
          this.$message.success('修改成功');
          this.key = new Date().getTime();//避免key重复
        }).catch(error => console.log(error))
      },
```

5.2 mingrui-shop-service-api-xxx

```
    @ApiOperation(value = "修改分类")
    @PutMapping(value = "category/edit")
    public Result<JSONObject> editCategory(@RequestBody CategoryEntity entity);

```

5.3 mingrui-shop-service-xxx

```
    @Override
    public Result<JSONObject> editCategory(CategoryEntity entity) {

        try {
            categoryMapper.updateByPrimaryKeySelective(entity);
        } catch (Exception e) {
            e.printStackTrace();
        }

        return this.setResultSuccess();
    }
```

6 商品分类管理(新增)

6.1 vue项目

6.1.1 Category.vue

```
        handleAdd(node) {

          //处理boolean值
          node.isParent = node.isParent?1:0;
          //删除id属性
          delete node.id;
          //新增
          this.$http.post('/category/add',node).then(resp => {
            this.$message.success('新增成功');
            this.key = new Date().getTime();//避免key重复
          }).catch(error => console.log(error))
        },
```

6.2 mingrui-shop-service-api-xxx

```
    @ApiOperation(value = "新增分类")
    @PostMapping(value = "category/add")
    public Result<JSONObject> addCategory(@RequestBody CategoryEntity entity);
```

6.3 mingrui-shop-service-xxx

```
    @Override
    public Result<JSONObject> addCategory(CategoryEntity entity) {

        try {
            categoryMapper.insertSelective(entity);
        } catch (Exception e) {
            e.printStackTrace();
        }

        return this.setResultSuccess();
    }
```

7 项目中不完美的地方

7.1 参数校验

7.1.1 common-core工程新建包com.baidu.shop.validate.group

7.1.1.1 包下新建操作类MingruiOperation

```
/**
 * @ClassName MingruiOperation
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
public class MingruiOperation {

    public interface Add{}
    public interface Update{}
}
```

7.1.2 mingrui-shop-service-api-xxx

7.1.2.1 entity

```

import com.baidu.shop.validate.group.MingruiOperation;
import io.swagger.annotations.ApiModel;
import io.swagger.annotations.ApiModelProperty;
import lombok.Data;

import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.NotEmpty;
import javax.validation.constraints.NotNull;

/**
 * @ClassName CategoryEntity
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@ApiModel(value = "分类实体类")
@Data
@Table(name = "tb_category")
public class CategoryEntity {

    @Id
    @ApiModelProperty(value = "分类主键",example = "1")
    //此处要引出来一个分组的概念,就是当前参数校验属于哪个组
    //有的操作不需要验证次参数就比如说新增就不需要校验id,但是修改需要
    @NotNull(message = "ID不能空",groups = {MingruiOperation.Update.class})
    private Integer id;

    @ApiModelProperty(value = "分类名称")
    //新增和修改都需要校验此参数
    @NotEmpty(message = "分类名称不能为空",groups = {MingruiOperation.Add.class,
MingruiOperation.Update.class})
    private String name;

    @ApiModelProperty(value = "父级分类",example = "1")
    @NotNull(message = "父级分类不能为null",groups = {MingruiOperation.Add.class})
    private Integer parentId;

    @ApiModelProperty(value = "是否是父级节点",example = "1")
    @NotNull(message = "是否为父级节点不能为null",groups =
{MingruiOperation.Add.class})
    private Integer isParent;

    @ApiModelProperty(value = "排序",example = "1")
    @NotNull(message = "排序字段不能为null",groups = {MingruiOperation.Add.class})
    private Integer sort;
}
```

7.1.2.2 service

```
import com.alibaba.fastjson.JSONObject;
import com.baidu.shop.base.Result;
import com.baidu.shop.entity.CategoryEntity;
import com.baidu.shop.validate.group.MingruiOperation;
import com.google.gson.JsonObject;
import io.swagger.annotations.Api;
import io.swagger.annotations.ApiOperation;
import org.springframework.validation.annotation.Validated;
import org.springframework.web.bind.annotation.*;

import java.util.List;

/**
 * @ClassName Category
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@Api(tags = "商品分类接口")
public interface CategoryService {

    @ApiOperation(value = "通过查询商品分类")
    @GetMapping(value = "category/list")
    public Result<List<CategoryEntity>> getCategoryByPid(Integer pid);

    @ApiOperation(value = "通过id删除分类")
    @DeleteMapping(value = "category/del")
    public Result<JsonObject> delCategory(Integer id);

    @ApiOperation(value = "修改分类")
    @PutMapping(value = "category/edit")
    //声明哪个组下面的参数参加校验-->当前是校验Update组
    public Result<JSONObject>
editCategory(@Validated({MingruiOperation.Update.class})@RequestBody
CategoryEntity entity);

    @ApiOperation(value = "新增分类")
    @PostMapping(value = "category/add")
    //声明哪个组下面的参数参加校验-->当前是校验新增组
    public Result<JSONObject>
addCategory(@Validated({MingruiOperation.Add.class}) @RequestBody CategoryEntity
entity);
}
```

7.2 全局异常处理

7.2.1 common-core

7.2.1.1 pom.xml

```
        <!-- SpringBoot-整合Web组件 -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
```

7.2.1.2 HTTPStatus

```
public static final int PARAMS_VALIDATE_ERROR = 5002;//参数校验失败
```

7.2.1.3 新建包com.baidu.shop.global

7.2.1.4 新建全局异常处理类GlobalException

```
import com.alibaba.fastjson.JSONObject;
import com.baidu.shop.base.Result;
import com.baidu.shop.status.HTTPStatus;
import com.google.gson.JsonObject;
import lombok.extern.slf4j.Slf4j;
import org.springframework.validation.FieldError;
import org.springframework.web.bind.MethodArgumentNotValidException;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.RestControllerAdvice;

import javax.servlet.http.HttpServletRequest;
import java.util.ArrayList;
import java.util.List;


/**
 * @ClassName GlobalException
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@RestControllerAdvice
@Slf4j
public class GlobalException {

    @ExceptionHandler(RuntimeException.class)
    public Result<JSONObject> test(HttpServletRequest req, Exception e){

        Result<JSONObject> result = new Result();
        result.setCode(HTTPStatus.ERROR);
        result.setMessage(e.getMessage());

        log.debug(e.getMessage());

        return result;
    }

    @ExceptionHandler(value=MethodArgumentNotValidException.class)
    public List<Result<JsonObject>>
MethodArgumentNotValidHandler(HttpServletRequest request,

MethodArgumentNotValidException exception) throws Exception
    {
        List<Result<JsonObject>> objects = new ArrayList<>();

        //按需重新封装需要返回的错误信息

        for (FieldError error : exception.getBindingResult().getFieldErrors()) {

            Result<JsonObject> jsonObjectResult = new Result<>();

            jsonObjectResult.setCode(HTTPStatus.PARAMS_VALIDATE_ERROR);

            jsonObjectResult.setMessage("Field --> " + error.getField() + " : "
+ error.getDefaultMessage());

            log.debug("Field --> " + error.getField() + " : " +
error.getDefaultMessage());
            objects.add(jsonObjectResult);
        }

        return objects;
    }
}
```

7.3 swagger

7.3.1 mingrui-shop-service-api

7.3.1.1 pom.xml

```
 <!--帮助开发人员快速生成API文档-->
        <dependency>
            <groupId>io.springfox</groupId>
            <artifactId>springfox-swagger2</artifactId>
            <version>2.9.2</version>
        </dependency>
        <!--提供可视化的API文档-->
        <dependency>
            <groupId>io.springfox</groupId>
            <artifactId>springfox-swagger-ui</artifactId>
            <version>2.9.2</version>
        </dependency>
```

7.3.2 mingrui-shop-service-api-xxx

在com.baidu.shop下新建config包

在config包下新建MrSwagger2Config

```
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import springfox.documentation.builders.ApiInfoBuilder;
import springfox.documentation.builders.PathSelectors;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.service.ApiInfo;
import springfox.documentation.service.Contact;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;
/**
 * @ClassName MrSwagger2Config
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/17
 * @Version V1.0
 **/
@Configuration
@EnableSwagger2
public class MrSwagger2Config {

    @Bean
    public Docket createRestApi(){
        return new Docket(DocumentationType.SWAGGER_2)
                .apiInfo(this.apiInfo())
                .select()
                .apis(RequestHandlerSelectors.basePackage("com.baidu"))
                .paths(PathSelectors.any())
                .build();
    }

    private ApiInfo apiInfo(){
        return new ApiInfoBuilder()
                //标题
                .title("明瑞SWAGGER2标题")
                //条款地址
                .termsOfServiceUrl("http://www.baidu.com")
                //联系方式-->有String参数的方法但是已经过时，所以不推荐使用
                .contact(new
Contact("shenyaqi","baidu.com","shenyaqiii@163.com"))
                //版本
                .version("v1.0")
                //项目描述
                .description("描述")
                //创建API基本信息
                .build();
    }
}
```