**‌**

**湘潭大学**

**2025信息系统实践课程**

**软件说明书‌**

**项目名称**‌：

‌**学 院**：计算机学院

‌**小组序号**‌：3

‌**成员姓名**‌：彭孝乐，王如月，曾洁，李宁

**指导老师**‌：尹兆远

‌**提交日期**‌：2025年5月15日

**‌**

**一、项目概述‌**

1.‌**项目背景**‌  
 随着电子商务的快速发展，商家对后台管理系统的需求日益增加。 本项目旨在开发一个功能完善的后台管理系统，帮助商家高效管理电商 平台的商品、用户等核心业务数据，提升运营效率和管理能力。 目标用户为小型电商企业的运营人员和管理员。

1. ‌**系统目标**‌  
   （说明系统实现的核心功能和预期效果）

本系统旨在实现以下核心功能：

商品添加：添加商品信息。

信息编辑：编辑商品信息。

    商品下架：下架删除商品信息。

富文本编辑器：实现富文本编辑。

图片上传：上传图片功能

商品查询：搜索查询商品信息。

国际化：多语言面向国际。

用户管理：管理用户账户登录登出。

规格管理：查看数据和修改参数。

2.‌**开发环境**‌  
 开发工具：Visual Studio Code

编程语言：JavaScript、HTML、CSS

框架：Vue.js、Element-UI

数据库：MySQL

服务器：Node.js + Express

版本控制：Git

**‌二、需求分析‌**

1. **功能需求**

1. 用户管理模块：支持用户使用账号密码登录，登出。
2. 商品管理模块：支持商品的增删改查、分类管理、图片上传。
3. 用户查询模块：支持按名字，描述查询商品。
4. 国际化模块：支持中英文切换。
5. 规格参数模块：支持对类目设定规格参数

2. **非功能需求**

1. 性能需求：系统响应时间小于1秒，支持1000+并发用户。
2. 安全性需求：支持用户账号密码登录，敏感数据加密存储。
3. 兼容性需求：支持主流浏览器（Chrome、Firefox、Edge）

**‌三、系统设计‌**

1.‌**系统架构**‌  
本系统采用B/S架构，前端使用Vue.js开发，后端使用Node.js + Express框架，数据库采用MySQL存储数据。

2.‌**模块设计**‌  
（按功能模块拆分，说明各模块职责及交互关系）

用户管理模块：负责用户的注册、登录、权限分配。

商品管理模块：负责商品的增删改查、分类管理、图片上传。

订单管理模块：负责订单的查询、状态更新、导出功能。

数据统计模块：负责销售数据的图表展示和趋势分析。

国际化模块：负责多语言切换功能。

3.‌**数据库设计**‌

oE-R图

o主要数据表结构（表格示例：字段名、类型、说明）

o

**‌四、系统实现‌**

1.‌**关键技术**‌

前端框架：Vue.js，用于构建用户界面。

后端框架：Express，用于处理API请求。

数据库：MySQL，用于存储系统数据。

国际化：使用Vue I18n实现多语言支持。

2.‌**界面展示**‌  
（提供主要功能界面截图并配文字说明）

3.‌**核心代码片段**‌  
（展示关键功能的代码并注释，例如数据库连接、核心算法）

Mysql:

//连接数据库 1 安装mysql 2创建连接

const mysql=require('mysql')

//创建数据库连接

const client =mysql.createConnection({

    host:'localhost',//数据域名 地址//

    user:'root',//数据名称

    password:'Zj243257607#',//数据库密码 xampp集成

    database:'vue-ego',

    port:'3316'

})

//封装数据库操作语言 sql语句 参数数组arr callback成功函数结果

function sqlFun(sql,arr,callback){

    client.query(sql,arr,function(error,result){

        if(error){

            console.log('数据库语句错误',error);

            return ;

        }

        callback(result)

    })

}

module.exports=sqlFun

Router:

const express = require('express')

const router = express.Router()

//导入数据库 sqlFn('sql',[],res=>{})

const sqlFn = require('./mysql')

//图片需要的模块

const multer = require('multer')

const fs = require('fs')

//导入模块 jsonwebtoken 密钥

const jwt=require('jsonwebtoken');

//config.jwtSecret

const config=require('./secret');

//登录接口

/\*\*

 \* 语法：

 \* 如60，"2 days"，"10h"，"7d"，Expiration time，过期时间

 \*  jwt.sign({},'秘钥','过期时间,{expiresIn:20\*1,'1day''1h'}')

 \*/

/\*\*

 \* 登录 login

 \* 接受的字段：username,password

 \* 测试：postman

 \*/

router.post('/login', (req, res) => {

    let { username, password } = req.body

    //请求数据库

    let sql = "select \* from user where username=? and password=?";

    let arr = [username, password]

    sqlFn(sql, arr, result => {

        if (result.length > 0) {

            let token = jwt.sign({

                username: result[0].username,

                id: result[0].id

            }, config.jwtSecret, {

                expiresIn: 20\*1

            })

            res.send({

                status: 200,

                data: token

            })

        } else {

            res.send({

                status: 404,

                msg: '信息错误'

            })

        }

    })

})

/\*\*

 \* 注册接口 /register

 \*/

router.post("/register", (req, res) => {

    const {

        username,

        password

    } = req.body;

    const sql = "insert into userinfo values(null,?,?)";

    const arr = [username, password];

    sqlFn(sql, arr, (result) => {

        if (result.affectedRows > 0) {

            res.send({

                msg: "注册成功",

                status: 200

            })

        } else {

            res.status(401).json({

                errors: "用户名密码错误"

            })

        }

    })

})

/\*\*

 \* 商品列表：获取分页 {total:'',arr:[{},{},{}],pagesize:8,}

 \* 参数：page 页码

 \*/

router.get('/projectList', (req, res) => {

    const page = req.query.page || 1;

    const sqlLen = "select \* from project where id";

    sqlFn(sqlLen, null, data => {

        let len = data.length;

        const sql = "select \* from project order by id desc limit 8 offset " + (page - 1) \* 8;

        sqlFn(sql, null, result => {

            if (result.length > 0) {

                res.send({

                    status: 200,

                    data: result,

                    pageSize: 8,

                    total: len

                })

            } else {

                res.send({

                    status: 500,

                    msg: "暂无数据"

                })

            }

        })

    })

})

/\*\*

 \* 商品查询接口 search

 \* 参数：search

 \*/

router.get("/search", (req, res) => {

    var search = req.query.search;

    const sql = "select \* from project where concat(`title`,`sellPoint`,`descs`) like '%" + search + "%'";

    sqlFn(sql, null, (result) => {

        if (result.length > 0) {

            res.send({

                status: 200,

                result

            })

        } else {

            res.send({

                status: 500,

                msg: "暂无数据"

            })

        }

    })

})

/\*\*

 \* 类目选择

 \* 接口说明：接口不同的参数cid 返回不同的类目数据 后台接受变量：id

 \*/

router.get("/backend/itemCategory/selectItemCategoryByParentId", (req, res) => {

    const id = req.query.id || 1;

    const sql = "select \* from category where id=?"

    var arr = [id];

    sqlFn(sql, arr, result => {

        if (result.length > 0) {

            res.send({

                status: 200,

                result

            })

        } else {

            res.send({

                status: 500,

                msg: "暂无数据"

            })

        }

    })

})

/\*\*

 \* 类目结构数据获取

 \*/

router.get("/category/data", (req, res) => {

    var cid = req.query.cid;

    var sql = "select \* from params where itemCatId=?";

    sqlFn(sql, [cid], result => {

        if (result.length > 0) {

            res.send({

                status: 200,

                result

            })

        } else {

            res.send({

                status: 500,

                msg: "暂无数据"

            })

        }

    })

})

/\*\*

 \* 上传图片 post请求 upload

 \* 说明：

 \*  1.后台安装 multer 模块   同时引入fs模块

 \*  2.router.js入口文件导入模块

 \*      const fs=require('fs')

        const multer=require('multer')

 \*  3.上传图片

 \*

 \*/

var storage = multer.diskStorage({

    destination: function (req, file, cb) {

        cb(null, "./upload/")

    },

    filename: function (req, file, cb) {

           cb(null, Date.now() + "-" + file.originalname)

    }

})

var createFolder = function (folder) {

    try {

        fs.accessSync(folder);

    } catch (e) {

        fs.mkdirSync(folder);

    }

}

var uploadFolder = './upload/';

createFolder(uploadFolder);

var upload = multer({

    storage: storage

});

router.post('/upload', upload.single('file'), function (req, res, next) {

    var file = req.file;

    console.log('文件类型：%s', file.mimetype);

    console.log('原始文件名：%s', file.originalname);

    console.log('文件大小：%s', file.size);

    console.log('文件保存路径：%s', file.path);

    res.json({

         res\_code: '0',

        name: file.originalname,

        url: file.path

    });

});

/\*\*

 \* 商品添加

 \* 参数： title cid  category sellPoint price num descs paramsInfo image

 \*/

router.get("/backend/item/insertTbItem", (req, res) => {

    /\*\*

     \* 获取参数

     \*/

    var title = req.query.title || "";

    var cid = req.query.cid || "";

    var category = req.query.category || "";

    var sellPoint = req.query.sellPoint || "";

    var price = req.query.price || "";

    var num = req.query.num || "";

    var desc = req.query.descs || "";

    var paramsInfo = req.query.paramsInfo || "";

    var image = req.query.image || "";

    const sql = "insert into project values (null,?,?,?,?,?,?,?,'',1,'','',?)"

    var arr = [title, image, sellPoint, price, cid,category, num, desc];

    sqlFn(sql, arr, result => {

        console.log(123);

        if (result.affectedRows > 0) {

            res.send({

                status: 200,

                msg: "添加成功"

            })

        } else {

            res.send({

                status: 500,

                msg: "添加失败"

            })

        }

    })

})

/\*\*

 \* 商品删除接口 id

 \*/

router.get("/backend/item/deleteItemById", (req, res) => {

    var id = req.query.id;

    const sql = "delete from project where id=?"

    const arr = [id];

    sqlFn(sql, arr, result => {

        if (result.affectedRows > 0) {

            res.send({

                status: 200,

                msg: "删除成功"

            })

        } else {

            res.send({

                status: 500,

                msg: "删除失败"

            })

        }

    })

})

/\*\*

 \* 批量删除 batchDelete  idArr Array  id标识

 \* sql = "delete from A where id in (1,2,3)"

 \*/

router.get("/batchDelete", (req, res) => {

    let arr = req.query.idArr;//[] 数组格式 需要传递数据是 离散的数字格式

    // const sql =`delete from project where id in (?)`;

    let sql = '';

    function fun(arr) {//sql =`delete from project where id in (101,102,103)`;

        sql = `delete from project where id in (`

        for (let i = 0; i < arr.length; i++) {

            sql += arr[i] + ',' //101,102,

        }

        sql = sql.slice(0, -1)

        sql = sql + ')'

        // console.log(sql);

    }

    fun(arr)

    sqlFun(sql, null, result => {

        if (result.affectedRows > 0) {

            res.send({

                status: 200,

                msg: "删除成功"

            })

        } else {

            res.send({

                status: 500,

                msg: "删除失败"

            })

        }

    })

})

/\*\*

 \* 修改商品

 \*/

router.get("/backend/item/updateTbItem", (req, res) => {

    var id = req.query.id;

    var title = req.query.title || "";

    var sellPoint = req.query.sellPoint || "";

    var price = req.query.price || "";

    var cid = req.query.cid || "";

    var category = req.query.category || "";

    var num = req.query.num || "";

    var desc = req.query.descs || "";

    var paramsInfo = req.query.paramsInfo || "";

    var image = req.query.image || "";

    var sql = "update project set title=?,sellPoint=?,price=?,cid=?,category=?,num=?,descs=?,image=? where id=?";

    var arr = [title, sellPoint, price, cid, category, num, desc, image, id];

    sqlFn(sql, arr, result => {

        if (result.affectedRows > 0) {

            res.send({

                status: 200,

                msg: "修改成功"

            })

        } else {

            res.send({

                status: 500,

                msg: "修改失败"

            })

        }

    })

})

/\*\*

 \* 规格参数列表  参数 page

 \*/

router.get("/backend/itemParam/selectItemParamAll", (req, res) => {

    const page = req.query.page || 1;

    const sqlLen = "select \* from params where id";

    sqlFn(sqlLen, null, data => {

        let len = data.length;

        const sql = "select \* from params order by id desc limit 8 offset " + (page - 1) \* 8;

        sqlFn(sql, null, result => {

            if (result.length > 0) {

                res.send({

                    status: 200,

                    data: result,

                    pageSize: 8,

                    total: len

                })

            } else {

                res.send({

                    status: 500,

                    msg: "暂无数据"

                })

            }

        })

    })

})

/\*\*

 \* 规格参数  模糊查询  参数：search

 \*/

router.get("/params/search", (req, res) => {

    var search = req.query.search;

    const sql = "select \* from params where concat(`paramData`) like '%" + search + "%'";

    sqlFn(sql, [search], result => {

        if (result.length > 0) {

            res.send({

                status: 200,

                result

            })

        } else {

            res.send({

                status: 500,

                msg: '暂无数据'

            })

        }

    })

})

/\*\*

 \* 规格参数 添加

 \* 参数：itemCatId,content,specsName

 \*/

router.get("/backend/itemParam/insertItemParam", (req, res) => {

    var itemCatId = req.query.itemCatId;

    var paramsContent = req.query.content;

    var specsName = req.query.specsName;

    // console.log(itemCatId, paramsContent,specsName);

    var sql = "insert into params values (null,?,?,?)";

    sqlFn(sql, [itemCatId,specsName ,paramsContent ], result => {

        if (result.affectedRows > 0) {

            res.send({

                status: 200,

                msg: "添加成功"

            })

        } else {

            res.send({

                status: 500,

                msg: "添加失败"

            })

        }

    })

})

/\*\*

 \* 修改规格参数 cid content id  specsName

 \*/

router.get("/update/category", (req, res) => {

    var cid = req.query.cid;

    var content = req.query.content;

    var id = req.query.id;

    var specsName = req.query.specsName;

    var sql = "update params set paramData=? ,itemCatId=?,specsName=? where id=?";

    sqlFn(sql, [content, cid, specsName, id], result => {

        if (result.affectedRows > 0) {

            res.send({

                status: 200,

                msg: "修改成功"

            })

        } else {

            res.send({

                status: 500,

                msg: "修改失败"

            })

        }

    })

})

/\*\*

 \* 规格参数删除

 \*/

router.get("/params/delete", (req, res) => {

    var id = req.query.id;

    const sql = "delete from params where id=?"

    const arr = [id];

    sqlFn(sql, arr, result => {

        if (result.affectedRows > 0) {

            res.send({

                status: 200,

                msg: "删除成功"

            })

        } else {

            res.send({

                status: 500,

                msg: "删除失败"

            })

        }

    })

})

/\*\*

 \* 规格参数类目结构数据获取 cid

 \*/

router.get("/category/data", (req, res) => {

    var cid = req.query.cid;

    var sql = "select \* from params where itemCatId=?";

    sqlFn(sql, [cid], result => {

        if (result.length > 0) {

            res.send({

                status: 200,

                result

            })

        } else {

            res.send({

                status: 500,

                msg: "暂无数据"

            })

        }

    })

})

module.exports = router

**‌五、系统测试‌**

1.‌**测试方案**‌

单元测试：对各模块的功能进行独立测试。

集成测试：测试模块之间的交互是否正常。

用户测试：邀请目标用户进行实际操作测试。

1. ‌**测试用例与结果**‌

| **测试功能点** | **输入数据** | **预期结果** | **实际结果** |
| --- | --- | --- | --- |
| 用户登录 | 正确用户名密码 | 登录成功 | 登录成功 |
| 商品添加 | 商品名称、价格等 | 商品成功添加到数据库 | 商品成功添加到数据库 |
| 查询商品 | 正确的名称或者描述 | 商品成功被查询到 | 商品成功被查询到 |
| 添加修改规格参数 | 在规格参数页面添加需要  的规格参数，在商品添加  页面更改添加商品的规格参数信息 | 成功添加规格参数 | 成功添加规格参数 |

1. ‌**问题与改进**‌  
   （记录测试中发现的问题及优化措施）

问题：图片上传功能在某些浏览器中不兼容。

改进：优化图片上传逻辑，增加兼容性测试。

**‌六、用户手册‌**

1. ‌**安装部署说明**‌
2. 安装Node.js和MySQL。
3. 克隆项目代码：git clone <仓库地址>。
4. 安装依赖：npm install。
5. 安装插件：npm i wangeditor --save ，npm install vue-i18n -S等等具体可查看项目中README.md文件中详细介绍
6. 启动项目：npm run serve
7. ‌**操作指南**‌

登录：输入用户名和密码。

登出：按右上角登出按钮可以退出登录。

改变语言：在右上方可点击更换语言。

添加商品和上传图片：进入商品管理模块，填写商品信息并提交，点击上传图片区域可以上传图片文件。

下架商品：删除按钮。

编辑修改商品：编辑按钮修改商品信息

查询商品：搜索栏输入名字或商品介绍可查询商品。

添加修改规格参数：在规格参数页面添加需要的规格参数，然后即可在商品添加页面更改添加商品的规格参数信息。

**‌七、项目总结‌**

1.‌**成果总结**‌  
本项目实现了电商后台管理系统的核心功能，满足了商家的基本需求。

2.‌**不足与改进方向**‌

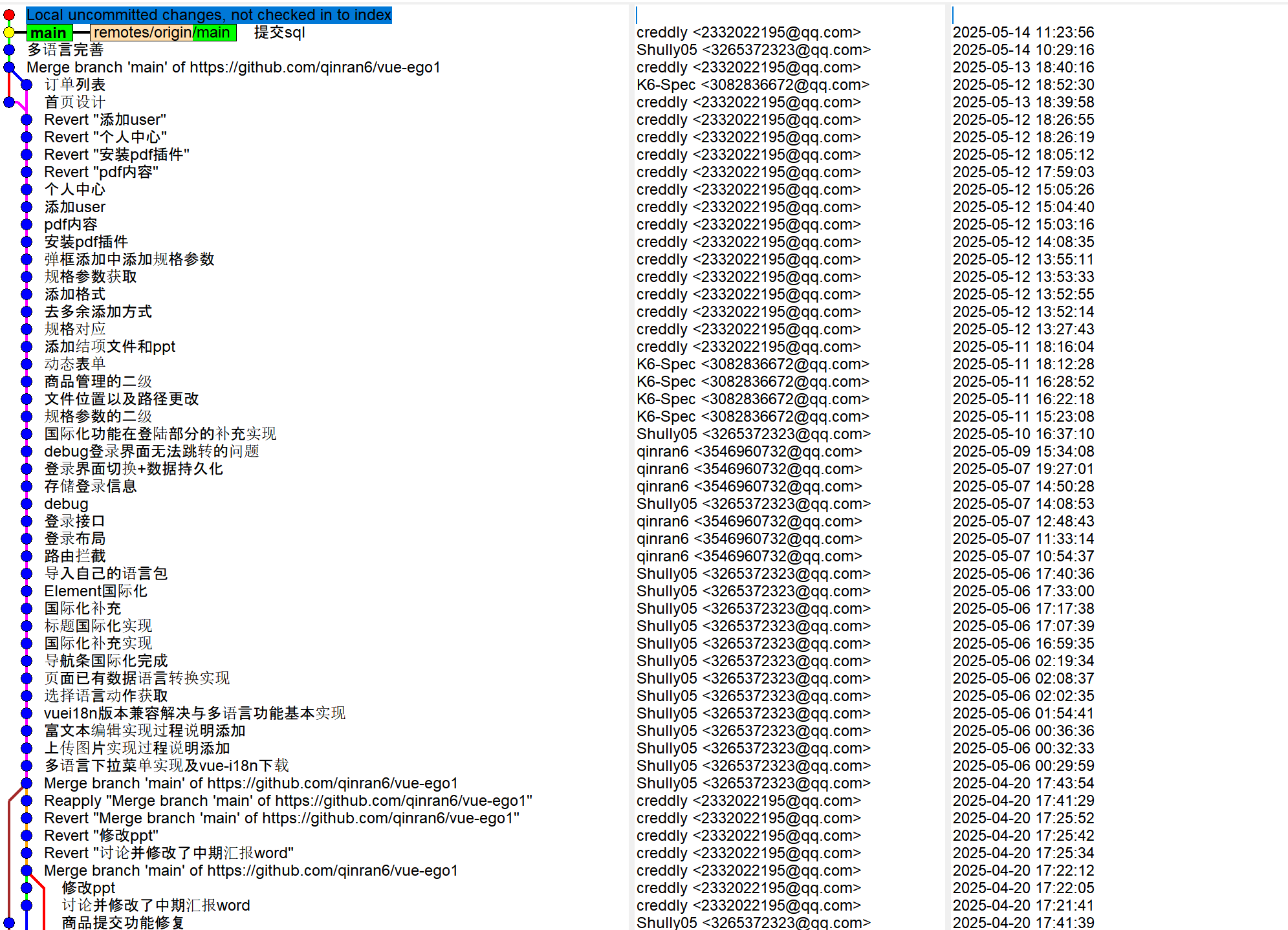
不足：部分功能（如高级数据分析和订单追踪等）尚未实现。

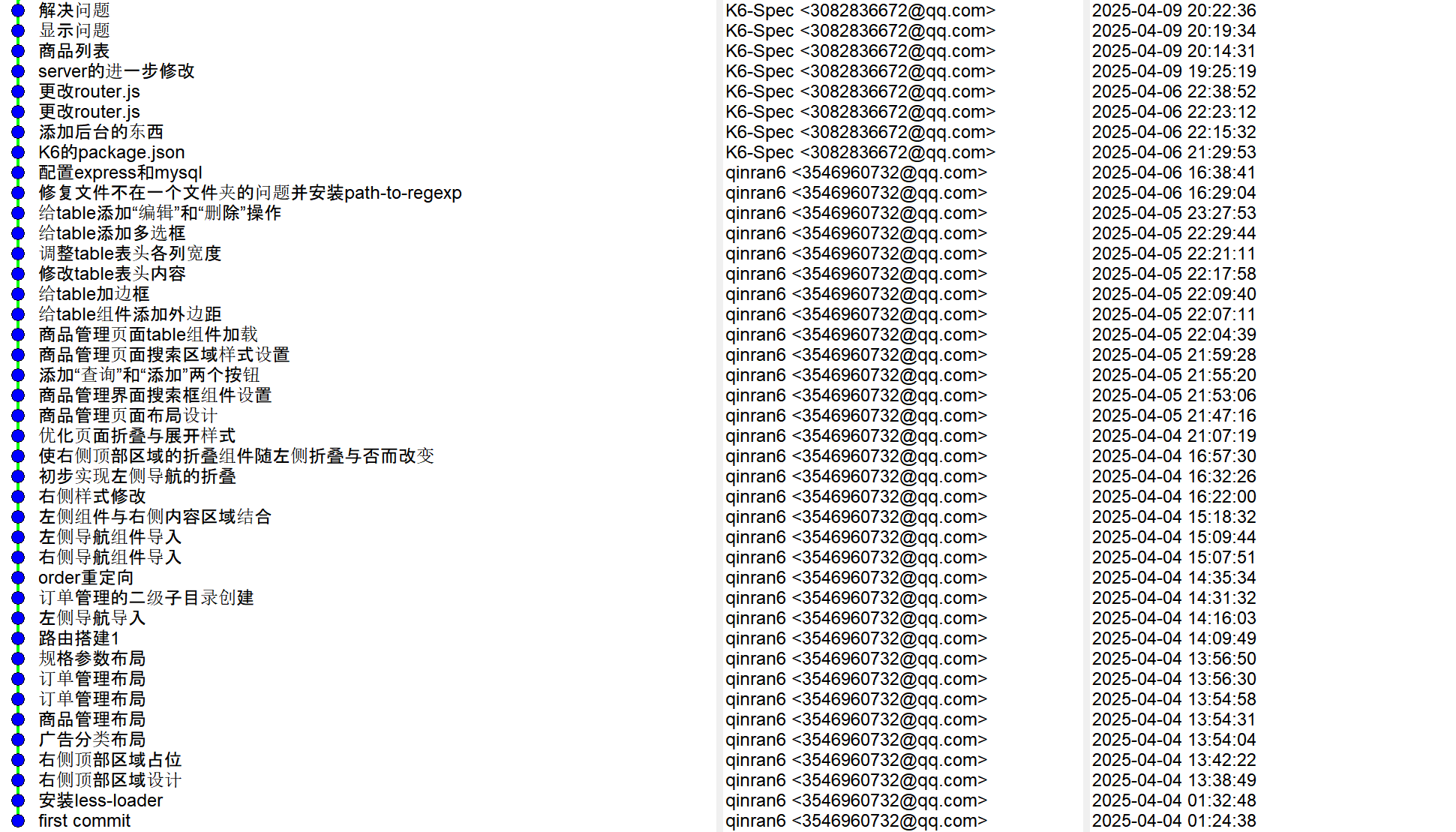
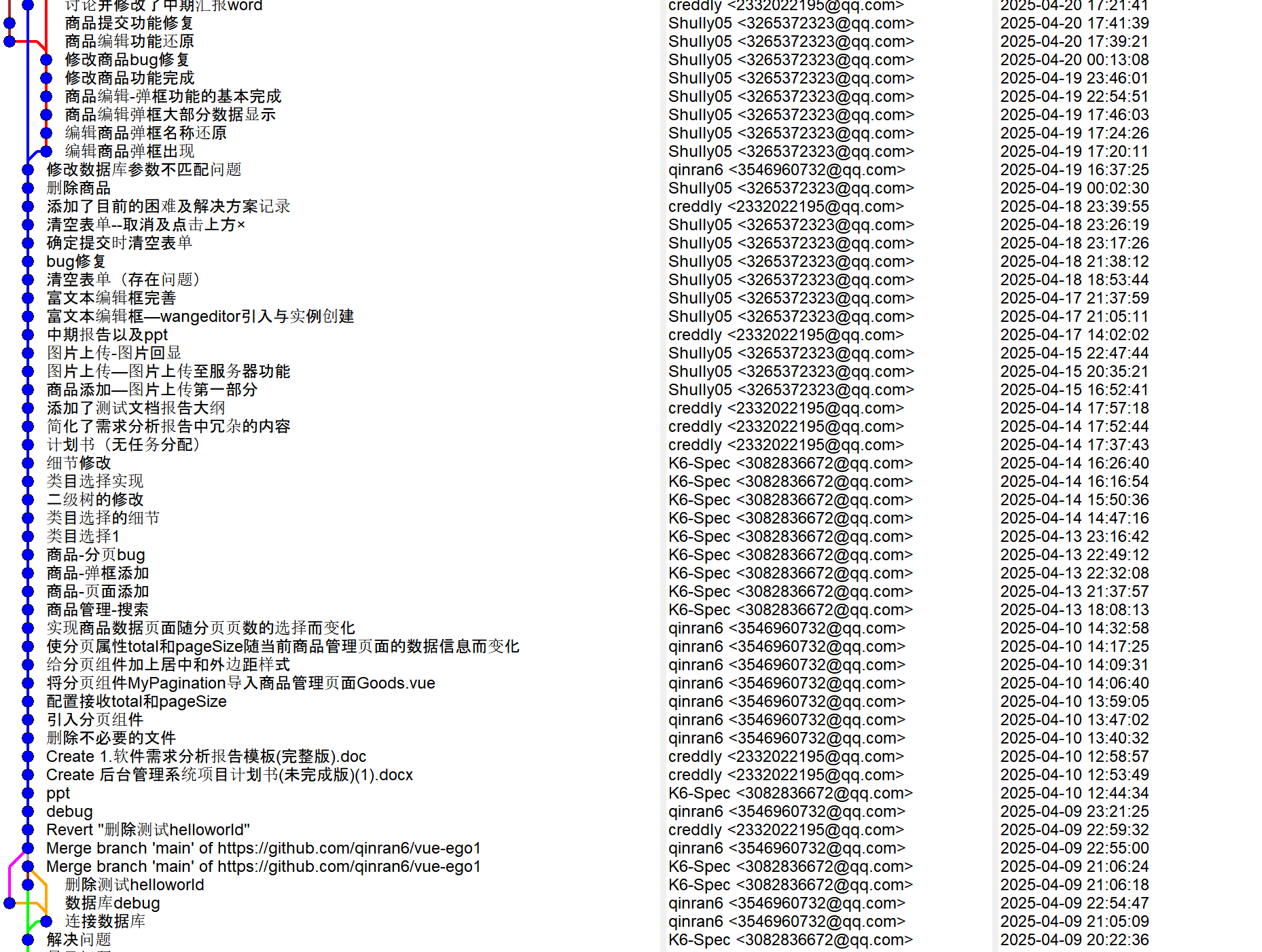
改进方向：未来可增加更多数据分析功能和便利功能，支持更多语言。

3.‌**成员分工表**‌  
（标明组长，说明各成员承担的具体任务，需要与Git提交记录对应）

| 成员姓名 | 分工内容 |
| --- | --- |
| 彭孝乐 | 商品图片上传、富文本编辑框、修改、删除功能实现、国际化多语言 |
| 王如月 | 首页版面设计实现，规格参数功能实现，测试，文档ppt编写完善 |
| 曾洁 | 项目框架、基础界面搭建及基础功能的实现，登陆布局及功能实现 |
| 李宁 | 商品的搜索、添加、分页以及对商品页面的进一步优化，规格页面  初始化 |

4.**Git提交记录**

**‌**



**附录‌**

* 参考资料（书籍、网站、开源项目等）

Vue.js官方文档：[https://vuejs.org/](vscode-file://vscode-app/c:/Users/23320/AppData/Local/Programs/Microsoft VS Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html" \o ")

Element-UI文档：[https://element.eleme.io/](vscode-file://vscode-app/c:/Users/23320/AppData/Local/Programs/Microsoft VS Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html" \o ")

MySQL官方文档：[https://dev.mysql.com/doc/](vscode-file://vscode-app/c:/Users/23320/AppData/Local/Programs/Microsoft VS Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html" \o ")

* 源代码仓库链接（如GitHub地址）

<https://github.com/qinran6/vue-ego1.git>

* 其他补充材料

无