**《前端开发体系结构》说明文档**

1. **学号：**

**姓名：**

题目：Shop系统

技术框架 Vue+Express+Mysql

Vue前端部分:使用Vue + VantUi

再测试期间调用了Mock，实现前端自检查

用户之间的交互功能在前端列出，共计15个请求，其余是数据表之间的关系，数据表有8个左右

文件夹 PATH 列表

├─build

├─config

├─middleware

├─node\_modules

├─src

│ ├─assets

│ │ ├─js

│ │ └─style

│ ├─components

│ ├─mock

│ ├─router

│ ├─utils

│ └─views

│ ├─catalog

│ ├─home

│ ├─live

│ ├─login

│ ├─my

│ ├─search

│ └─talk

└─static

### 界面功能：

首页





##### 关键代码

<template>

<div>

<header class="headerNav">

<img class="logo" src="../../assets/logo.png"/>

<div class="centerFix">

<router-link to="/search">

<van-search

style="height: 100%"

input-align="center"

v-model="value"

@click="goSearch"

disabled

placeholder="商品搜索"

/>

</router-link>

</div>

</header>

<main>

<div class="swiper">

<van-swipe class="my-swipe" :autoplay="3000">

<van-swipe-item

class="van-swipe-item"

v-for="(hot\_product, index) in images"

:key="hot\_product.product\_img"

>

<img v-lazy="hot\_product.product\_img"/>

</van-swipe-item>

</van-swipe>

</div>

<div class="advertise">

<van-grid :column-num="3" icon-size="10px">

<van-grid-item icon="smile" text="限时秒杀"/>

<van-grid-item icon="service" text="限量抢购"/>

<van-grid-item icon="friends" text="砍价聚惠"/>

</van-grid>

</div>

<div class="hotProduct">

<p class="desText">热销商品</p>

<van-grid clickable :column-num="2" square>

<van-grid-item

v-for="banner in hotBanner"

:key="banner.product\_img"

>

<van-image

:src="banner.product\_img"

/>

</van-grid-item>

</van-grid>

</div>

<p class="desText">爆款推荐</p>

<div

v-for="(info, index) in productList"

:key="info.product\_id"

class="productList"

>

<router-link to="/product-detail">

<van-card

:num="info.product\_number"

tag="火爆售卖"

:price="info.product\_price"

:desc="info.product\_desc"

:title="info.product\_name"

:thumb="info.product\_img"

/>

</router-link>

</div>

</main>

<!--底部导航 -->

<footer\_tabBar :active="0"></footer\_tabBar>

</div>

</template>

<script>

import footer\_tabBar from "../../components/footer-tabBar";

export default {

name: "home-index",

data() {

return {

value: "",

images: [],

productList: [],

hotBanner: []

};

},

components: {footer\_tabBar},

methods: {

goSearch() {

},

async setHotList() {

let token = localStorage.getItem("token");

let res = await this.$http.getHotList({token});

this.productList = res.list

// console.log(res)

},

async setBanner() {

let token = localStorage.getItem("token");

let res = await this.$http.getBanner({token});

// console.log(res)

this.images = res.list

},

async setHotBanner() {

let token = localStorage.getItem("token");

let res = await this.$http.getHotBanner({token});

console.log(res)

this.hotBanner = res.list

}

},

mounted() {

this.setHotList()

this.setBanner()

this.setHotBanner()

}

};

</script>

### 商品详情



##### 关键代码

<template>

<div>

<van-icon @click="onBackward" name="arrow-left"/>

<main>

<div class="banner-image">

<van-swipe class="my-swipe">

<van-swipe-item

class="van-swipe-item"

>

<img v-lazy="goods.product\_img"/>

</van-swipe-item>

</van-swipe>

</div>

<div class="info showboard">

<p class="price">￥{{ goods.product\_price }}</p>

<p class="title">{{ goods.product\_name }}</p>

</div>

<div class="divider"></div>

<div class="team">

<div class="team-title">

<p class="title" style="left: 0">加入拼团</p>

<p class="title" style="right: 0">更多</p>

</div>

<div class="team-list">

<van-image

v-for="user in userList"

round

:key="user.user\_id"

fit="cover"

width="1rem"

height="1rem"

:src="user.user\_avatar"

/>

</div>

</div>

<div class="divider"></div>

<div class="comment">

<div class="team-title">

<p class="title" style="left: 0">评价</p>

<p class="title" style="right: 0">查看全部</p>

</div>

<div class="comment-list"

v-for="comment in commentList"

:key="comment.comment\_id"

>

<van-image

round

fit="cover"

width="1rem"

height="1rem"

:src="comment.user\_avatar"

/>

<p class="comment">{{comment.comment\_info.comment\_content}}</p>

</div>

</div>

<div class="divider"></div>

<div class="detail">

<van-image

fit="cover"

width="100%"

height="100%"

v-for="item in detailImg"

:key="item"

:src="item"

/>

</div>

</main>

<footer>

<van-goods-action>

<van-goods-action-icon icon="chat-o" text="客服"/>

<van-goods-action-icon icon="shop-o" text="店铺"/>

<van-goods-action-button

color="#49B259"

type="warning"

text="单独购买"

/>

<van-goods-action-button

color="#69FF80"

type="danger"

text="发起拼团购买"

/>

</van-goods-action>

</footer>

</div>

</template>

<script>

export default {

name: "product-detailPage",

data() {

return {

goods: {},

images: [],

detailImg: [],

userList: [],

commentList: [],

};

},

mounted() {

this.setProductDetail()

this.setProductDetailComment()

this.setProductDetailGoods()

this.setProductUsers()

},

methods: {

onBackward() {

this.$router.go(-1);

},

async setProductDetail() {

let res = await this.$http.getProductDetail({});

// console.log(res)

this.goods = res.product\_info

},

async setProductDetailComment() {

let res = await this.$http.getProductDetailComment({});

console.log(res)

this.commentList = res.list

},

async setProductDetailGoods() {

let res = await this.$http.getProductDetailGoodsImg({});

this.detailImg = res.list

},

async setProductUsers() {

let res = await this.$http.getProductUsers({});

this.userList = res.list;

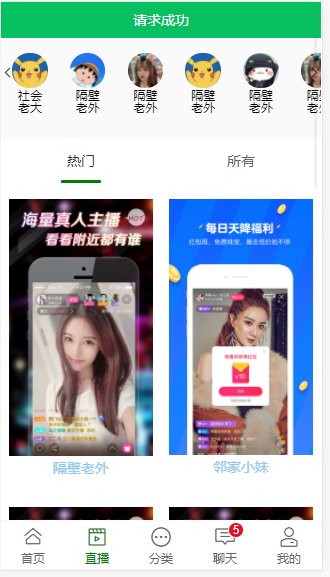
}

},

};

</script>

### 直播详情页面



此处采用获取所有的海报和Id，以及商户信息

<template>

<div class="body">

<header style="position: sticky; top: 0px">

<van-icon

v-show="storeList.length > 5"

class="moreStore"

name="arrow"

/>

<van-icon

v-show="storeList.length > 5"

class="moreStore"

style="left: 0"

name="arrow-left"

/>

<h2 style="text-align:center;font-size: 0.3rem; padding: 0.3rem">关注的直播</h2>

<div class="staredStore">

<div

class="storeComponent"

v-for="user in storeList"

:key="user.user\_id"

>

<van-image

round

fit="cover"

width="0.8rem"

height="0.8rem"

:src="user.user\_avatar"

/>

<p>{{ user.user\_name }}</p>

</div>

</div>

</header>

<main class="mainView">

<van-tabs color="green" v-model="active" swipeable sticky animated>

<van-tab

v-for="index in ['热门','所有']"

:key="index"

:title="index"

>

<van-grid :border="false" :column-num="2">

<van-grid-item v-for="live in live\_list" :key="live.live\_id" >

<router-link to="/live">

<van-image

:src="live.live\_poster"

/>

<h3 class="order">{{live.live\_name}}</h3>

</router-link>

</van-grid-item>

</van-grid>

</van-tab>

</van-tabs>

</main>

<footer>

<footer\_tabBar :active="1"></footer\_tabBar>

</footer>

</div>

</template>

<script>

import footer\_tabBar from "../../components/footer-tabBar";

export default {

name: "live-index",

data() {

return {

storeList: [],

active: "",

live\_list: []

};

},

components: {footer\_tabBar},

methods: {

async setStoreList() {

let res = await this.$http.getStoreList()

console.log(res)

this.storeList = res.list

},

async setLiveList() {

let res = await this.$http.getLiveList()

console.log(res)

this.live\_list = res.list

},

},

mounted() {

this.setStoreList()

this.setLiveList()

}

};

</script>

### 搜索界面



### 关键代码

<template>

<div>

<div class="search-bar">

<van-icon @click="onBackward" id="goback" name="arrow-left" />

<van-search

v-model="value"

show-action

placeholder="请输入搜索关键词"

@search="onSearch"

@clear="onClear"

>

<template #action>

<div @click="onSearch">搜索</div>

</template>

</van-search>

</div>

<van-dropdown-menu v-show="SEARCHED">

<van-dropdown-item v-model="value1" :options="option1" />

<van-dropdown-item v-model="value2" :options="option2" />

</van-dropdown-menu>

<div

v-for="(item, index) in productList"

:key="index"

class="productList"

>

<router-link to="/product-detail">

<van-card

:tag="item.tags[0]"

:price="item.price"

:desc="item.des"

:title="item.name"

thumb="https://img.yzcdn.cn/vant/ipad.jpeg"

:origin-price="item.originprice"

/>

</router-link>

</div>

</div>

</template>

<script>

import axios from "axios";

export default {

name: "search",

data() {

return {

value: "",

SEARCHED: false,

productList: [],

value1: 0,

value2: "a",

option1: [

{ text: "全部商品", value: 0 },

{ text: "新款商品", value: 1 },

{ text: "活动商品", value: 2 },

],

option2: [

{ text: "默认排序", value: "a" },

{ text: "好评排序", value: "b" },

{ text: "销量排序", value: "c" },

],

};

},

methods: {

onSearch() {

if (this.SEARCHED == false) this.SEARCHED = true;

axios

.get("http://localhost:3000/api1", {

params: {

pattern: this.value, //传参

},

})

.then((res) => {

// 直接修改索引项vue无法检测到

// for (let index in res.data) {

// this.productList[index] = res.data[index];

// }

console.log(res.data[0].name);

if (res.data[0].name == "还没有输入~") {

alert("还没有输入~");

}else if(res.data[0].name == "没有该商品信息~") {

alert("没有该商品信息~");

}

else {

this.productList = res.data;

}

console.log(this.productList);

});

},

onClear() {

this.value = "";

},

onBackward() {

//编程式导航

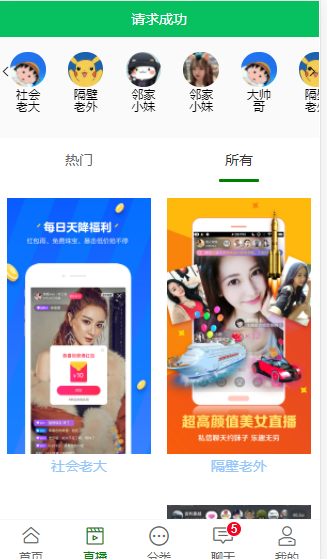
this.$router.go(-1);

},

},

};

</script>



### 关键代码：

<template>

<div class="body">

<header style="position: sticky; top: 0px">

<van-icon

v-show="storeList.length > 5"

class="moreStore"

name="arrow"

/>

<van-icon

v-show="storeList.length > 5"

class="moreStore"

style="left: 0"

name="arrow-left"

/>

<h2 style="text-align:center;font-size: 0.3rem; padding: 0.3rem">关注的直播</h2>

<div class="staredStore">

<div

class="storeComponent"

v-for="user in storeList"

:key="user.user\_id"

>

<van-image

round

fit="cover"

width="0.8rem"

height="0.8rem"

:src="user.user\_avatar"

/>

<p>{{ user.user\_name }}</p>

</div>

</div>

</header>

<main class="mainView">

<van-tabs color="green" v-model="active" swipeable sticky animated>

<van-tab

v-for="index in ['热门','所有']"

:key="index"

:title="index"

>

<van-grid :border="false" :column-num="2">

<van-grid-item v-for="live in live\_list" :key="live.live\_id" >

<router-link to="/live">

<van-image

:src="live.live\_poster"

/>

<h3 class="order">{{live.live\_name}}</h3>

</router-link>

</van-grid-item>

</van-grid>

</van-tab>

</van-tabs>

</main>

<footer>

<footer\_tabBar :active="1"></footer\_tabBar>

</footer>

</div>

</template>

<script>

import footer\_tabBar from "../../components/footer-tabBar";

export default {

name: "live-index",

data() {

return {

storeList: [],

active: "",

live\_list: []

};

},

components: {footer\_tabBar},

methods: {

async setStoreList() {

let res = await this.$http.getStoreList()

console.log(res)

this.storeList = res.list

},

async setLiveList() {

let res = await this.$http.getLiveList()

console.log(res)

this.live\_list = res.list

},

},

mounted() {

this.setStoreList()

this.setLiveList()

}

};

</script>

### 分类界面



关键代码：

<template>

<div>

<header>

<router-link to="/search">

<van-search

style="height: 100%"

input-align="center"

v-model="value"

@click="goSearch"

disabled

placeholder="商品搜索"

/>

</router-link>

</header>

<main>

<van-tree-select

height="100vh"

:items="items"

:main-active-index.sync="active"

>

<template #content>

<div v-for="(item,index) in product\_info" v-if="active === index">

<van-image

v-for="({product\_id,product\_info},index) in item.info"

:key="product\_id"

v-if="product\_info"

:src="product\_info.product\_img"

/>

</div>

</template>

</van-tree-select>

</main>

<footer>

<footer\_tabBar :active="2"></footer\_tabBar>

</footer>

</div>

</template>

<script>

import footer\_tabBar from "../../components/footer-tabBar";

export default {

name: "catalog-index",

data() {

return {

value: "",

active: 0,

items: [],

imgList: [{

product\_id: 1,

product\_img: "https://img.yzcdn.cn/vant/apple-3.jpg"

}],

product\_info:[],

timer:1

};

},

mounted() {

this.getItem()

},

methods: {

async getItem() {

let token = localStorage.getItem("token");

let res = await this.$http.getCatalog({token});

console.log(res)

this.items = res.item.map(val => {

return {text: val.type}

})

this.product\_info = res.item

console.log(this.items)

console.log(this.product\_info)

},

goSearch() {

// console.log("goSearch!");

},

},

components: {footer\_tabBar},

};

</script>





### 聊天页面



关键代码

<template>

<div>

<header>

<h4 style="text-align:center;font-size:28px;">

消息中心

</h4>

</header>

<main>

<van-pull-refresh v-model="refreshing" @refresh="onRefresh">

<van-list

v-model="loading"

:finished="finished"

finished-text="没有更多了"

@load="onLoad"

>

<div v-for="item in list"

class="chat-btn"

:key="item.user\_id"

@click="handleToTalkDetail"

>

<van-image

id="avatar"

round

fit="cover"

width="0.6rem"

height="0.6rem"

:src="item.user\_avatar"

/>

<span class="badge">{{ item.user\_name }}</span>

</div>

</van-list>

</van-pull-refresh>

</main>

<footer>

<footer\_tabBar :active="3"></footer\_tabBar>

</footer>

</div>

</template>

<script>

import footer\_tabBar from "../../components/footer-tabBar";

export default {

name: "talk-index",

data() {

return {

storename: ["咸鱼润", "下手"],

list: [],

loading: false,

finished: false,

refreshing: false,

};

},

mounted() {

this.setChatList()

},

methods: {

handleToTalkDetail(live\_id) {

this.$router.push({

path: "/talk-detail", meta: {

live\_id

}

})

},

async setChatList(){

let res =await this.$http.getChatList({})

console.log(res)

this.list = res.list

},

onLoad() {

setTimeout(() => {

if (this.refreshing) {

this.refreshing = false;

}

this.loading = false;

if (this.list.length >= 10) {

this.finished = true;

}

}, 1000);

},

onRefresh() {

// 清空列表数据

this.finished = false;

// 重新加载数据

// 将 loading 设置为 true，表示处于加载状态

this.loading = true;

this.onLoad();

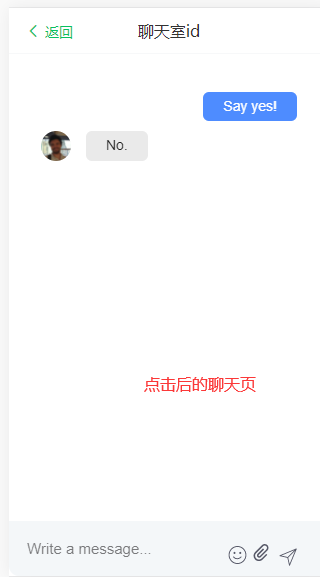
},

},

components: {footer\_tabBar},

};

</script>



关键代码

<template>

<section>

<header\_titleBar title="聊天室id"></header\_titleBar>

<main>

<beautiful-chat

class="sessionbody"

:participants="participants"

:titleImageUrl="titleImageUrl"

:onMessageWasSent="onMessageWasSent"

:messageList="messageList"

:newMessagesCount="newMessagesCount"

:isOpen="isChatOpen"

:close="closeChat"

:open="openChat"

:showEmoji="true"

:showFile="true"

:showEdition="true"

:showDeletion="true"

:deletionConfirmation="true"

:showTypingIndicator="showTypingIndicator"

:showLauncher="true"

:showCloseButton="true"

:colors="colors"

:alwaysScrollToBottom="alwaysScrollToBottom"

:disableUserListToggle="false"

:messageStyling="messageStyling"

@onType="handleOnType"

@edit="editMessage"

/>

</main>

</section>

</template>

<script>

import header\_titleBar from "./header-titileBar";

export default {

name: "talk-detail",

data() {

return {

participants: [

{

id: "user1",

name: "Matteo",

imageUrl:

"https://avatars3.githubusercontent.com/u/1915989?s=230&v=4",

},

{

id: "user2",

name: "Support",

imageUrl:

"https://avatars3.githubusercontent.com/u/37018832?s=200&v=4",

},

], // the list of all the participant of the conversation. `name` is the user name, `id` is used to establish the author of a message, `imageUrl` is supposed to be the user avatar.

titleImageUrl:

"https://a.slack-edge.com/66f9/img/avatars-teams/ava\_0001-34.png",

messageList: [

{type: "text", author: `me`, data: {text: `Say yes!`}},

{type: "text", author: `user1`, data: {text: `No.`}},

], // the list of the messages to show, can be paginated and adjusted dynamically

newMessagesCount: 0,

isChatOpen: true, // to determine whether the chat window should be open or closed

showTypingIndicator: "", // when set to a value matching the participant.id it shows the typing indicator for the specific user

colors: {

header: {

bg: "#4e8cff",

text: "#ffffff",

},

launcher: {

bg: "#4e8cff",

},

messageList: {

bg: "#ffffff",

},

sentMessage: {

bg: "#4e8cff",

text: "#ffffff",

},

receivedMessage: {

bg: "#eaeaea",

text: "#222222",

},

userInput: {

bg: "#f4f7f9",

text: "#565867",

},

}, // specifies the color scheme for the component

alwaysScrollToBottom: false, // when set to true always scrolls the chat to the bottom when new events are in (new message, user starts typing...)

messageStyling: true, // enables \*bold\* /emph/ \_underline\_ and such (more info at github.com/mattezza/msgdown)

};

},

mounted() {

this.fixedTitle();

},

methods: {

// 调整头部导航

fixedTitle() {

let titlebar = document.getElementsByClassName("sc-header")[0];

let titleoffset = document.getElementsByClassName(

"sc-chat-window"

)[0];

titleoffset.style = "padding-top: 1rem"

titlebar.style = "display: none";

},

sendMessage(text) {

if (text.length > 0) {

this.newMessagesCount = this.isChatOpen

? this.newMessagesCount

: this.newMessagesCount + 1;

this.onMessageWasSent({

author: "support",

type: "text",

data: {text},

});

}

},

onMessageWasSent(message) {

// called when the user sends a message

this.messageList = [...this.messageList, message];

},

openChat() {

// called when the user clicks on the fab button to open the chat

this.isChatOpen = true;

this.newMessagesCount = 0;

},

closeChat() {

this.isChatOpen = false;

},

handleScrollToTop() {

},

handleOnType() {

console.log("Emit typing event");

},

editMessage(message) {

const m = this.messageList.find((m) => m.id === message.id);

m.isEdited = true;

m.data.text = message.data.text;

},

},

components: {header\_titleBar},

};

</script>

<style scoped>

</style>

### 个人信息界面



关键代码：

<template>

<div>

<header>

<van-image

id="avatar"

round

fit="cover"

width="2rem"

height="2rem"

src="https://img.yzcdn.cn/vant/cat.jpeg"

/>

<label for="avatar">

{{ username }}<br/>

<van-icon name="edit"/>

</label>

<van-icon class="setting-van-icon" name="setting-o"/>

</header>

<main>

<div class="myOrderlist">

<div

style="position: relative; padding: 0 0.2rem 0.2rem 0.2rem"

>

<span class="moduleTitle">我的订单</span>

<span class="moduleTitle right">查看全部 ></span>

</div>

<van-grid :border="false" clickable :column-num="5">

<van-grid-item icon="home-o" text="待付款" to="/"/>

<van-grid-item icon="home-o" text="待分享" to="/"/>

<van-grid-item icon="home-o" text="待发货" to="/"/>

<van-grid-item icon="home-o" text="待收货" to="/"/>

<van-grid-item icon="home-o" text="待评价" to="/"/>

</van-grid>

</div>

<p class="divider"></p>

<div class="functionModule">

<van-grid clickable :column-num="2" square>

<van-grid-item icon="home-o" text="商品收藏" to="/"/>

<van-grid-item icon="home-o" text="历史浏览" to="/"/>

<van-grid-item icon="home-o" text="退款售后" to="/"/>

<van-grid-item icon="home-o" text="收货地址" to="/"/>

</van-grid>

</div>

<p class="divider"></p>

</main>

<footer>

<footer\_tabBar :active="4"></footer\_tabBar>

</footer>

</div>

</template>

<script>

import footer\_tabBar from "../../components/footer-tabBar";

export default {

name: "my-index",

data() {

return {

username: "",

};

},

mounted() {

this.username = localStorage.getItem("userOnlogin")

},

components: {footer\_tabBar},

};

</script>



#### 注册界面



关键代码：

<template>

<div>

<!-- 头部导航 -->

<header\_titleBar title="登录"></header\_titleBar>

<main>

<div class="logo">

<!--<img :src="logo" alt="logo" />-->

</div>

<!-- 注册表单 -->

<div v-show="!ONLOGIN" class="register-form">

<van-form @submit="onRegister">

<van-field

v-model="username"

type="text"

name="user\_name"

label="昵称"

placeholder="昵称"

:rules="[{ required: true, message: '请填写昵称' }]"

/>

<van-field

v-model="account"

type="text"

name="user\_account"

label="账户"

placeholder="设置账户"

:rules="[{ required: true, message: '请填写账户' }]"

/>

<van-field

v-model="password"

type="password"

name="user\_password"

label="密码"

placeholder="设置密码"

:rules="[{ required: true, message: '请填写密码' }]"

/>

<div style="margin: 0.2rem">

<van-button

:disabled="!isable"

type="primary"

round

block

native-type="submit"

>

注册

</van-button>

</div>

</van-form>

</div>

<!-- 登录表单 -->

<div v-show="ONLOGIN" class="login-form">

<van-form @submit="onLogin">

<van-field

v-model="username"

name="user\_account"

label="用户名"

placeholder="用户名"

:rules="[{ required: true, message: '请填写用户名' }]"

/>

<van-field

v-model="password"

type="password"

name="user\_password"

label="密码"

placeholder="密码"

:rules="[{ required: true, message: '请填写密码' }]"

/>

<div style="margin: 0.2rem">

<van-button

type="primary"

round

block

native-type="submit"

>

登录

</van-button>

<van-button

v-show="ONLOGIN"

@click="getRegister"

type="primary"

plain

block

native-type="button"

>注册

</van-button

>

</div>

</van-form>

</div>

</main>

</div>

</template>

<script>

import header\_titleBar from "../../components/header-titileBar";

export default {

name: "login-index",

data() {

return {

username: "",

emailaddress: "",

validatecode: "",

password: "",

account: "",

ONLOGIN: true,

isable: true,

};

},

methods: {

async onRegister(values) {

await this.asyncValidator("value");

let res = await this.$http.register(values)

console.log(res)

},

async onLogin(values) {

let res = await this.$http.postLogin(values)

localStorage.setItem("token", res.token)

let token = res.token;

let userinfo = await this.$http.userInfo({token})

localStorage.setItem("userOnlogin", userinfo.user\_name)

await this.$router.push({path: "/my"});

},

getRegister() {

this.ONLOGIN = false;

},

getValidateCode() {

alert("send");

},

asyncValidator(val) {

return new Promise((resolve) => {

this.$toast.loading("验证中...");

this.isable = false;

setTimeout(() => {

this.$toast.clear();

resolve(/\d{6}/.test(val));

this.isable = true;

}, 2000);

});

},

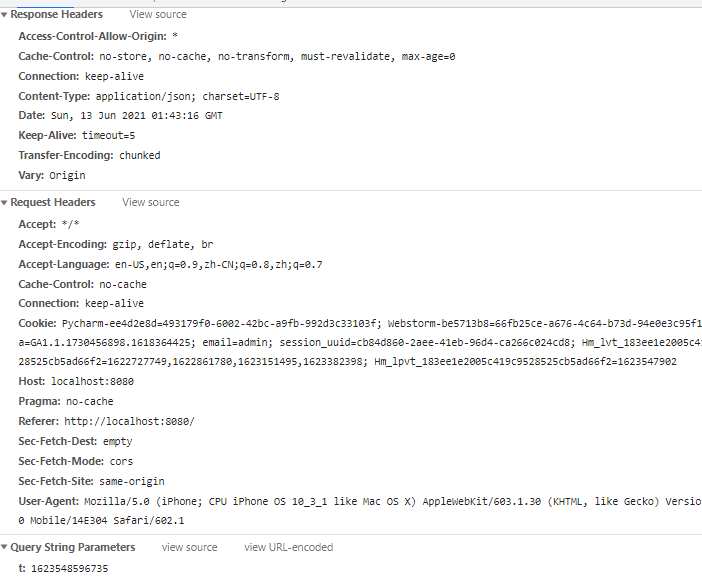
},

components: {header\_titleBar},

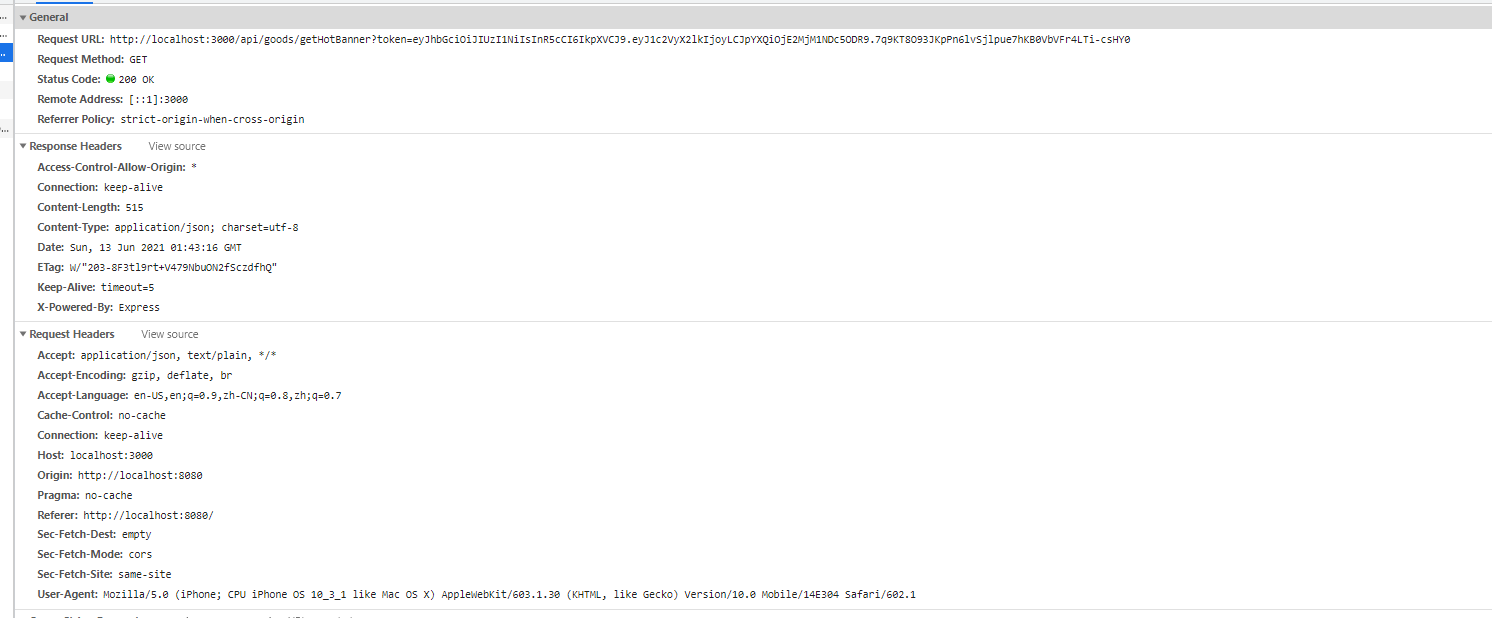
};

</script>

前端请求抓包









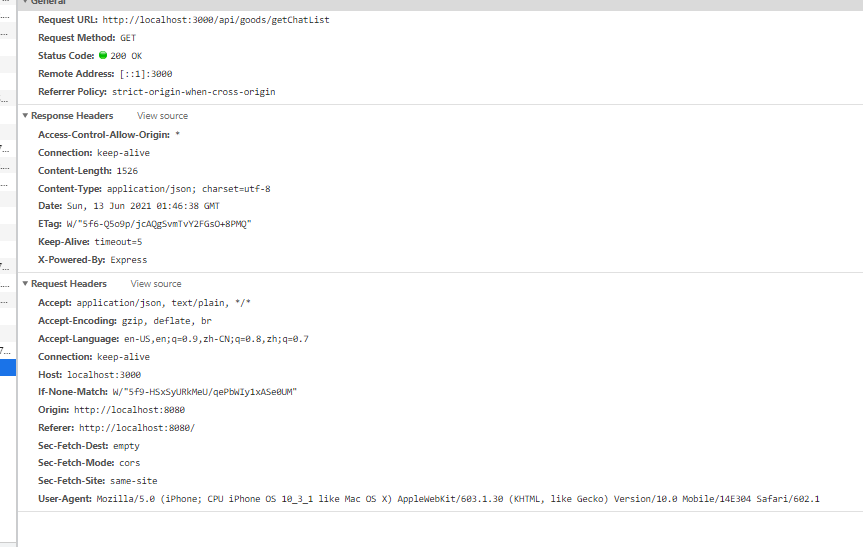


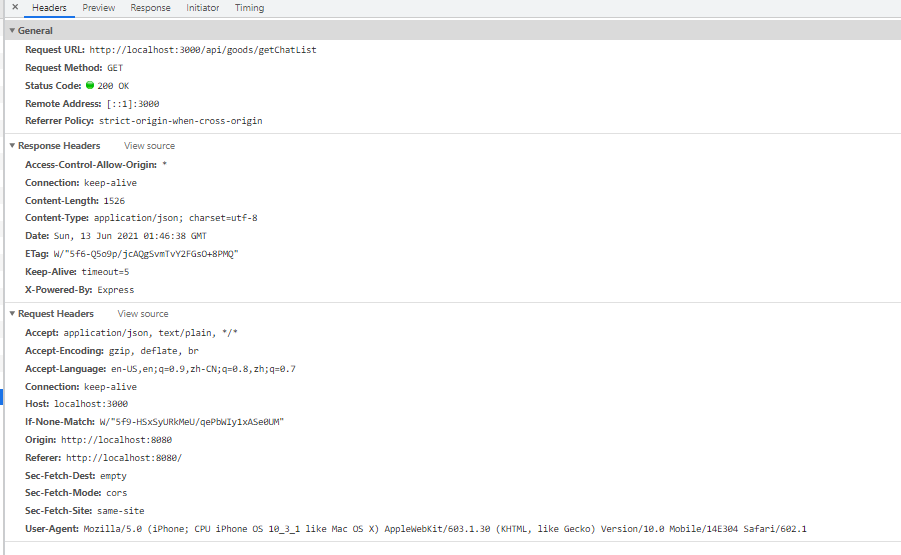


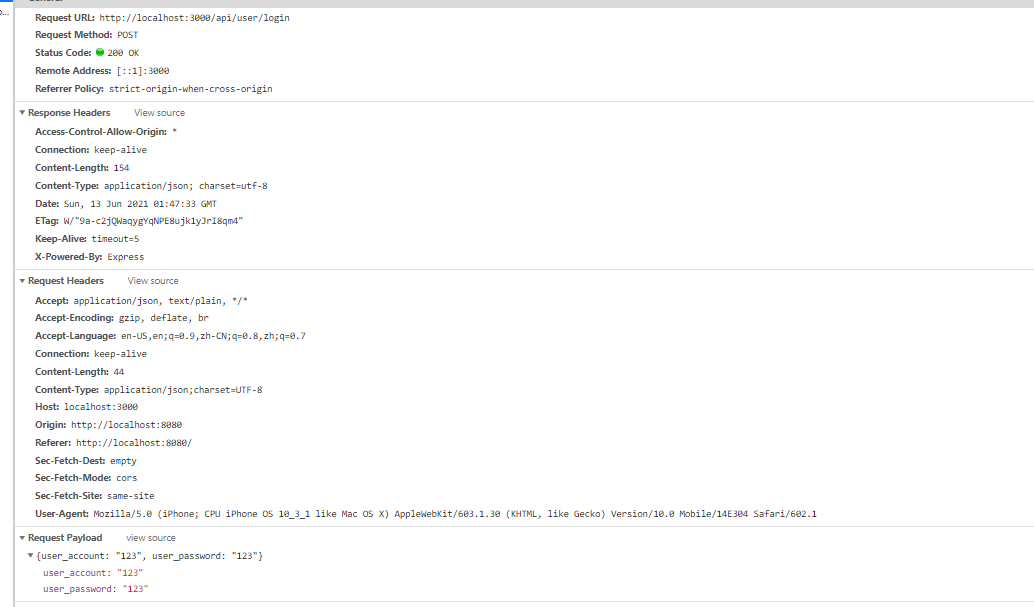


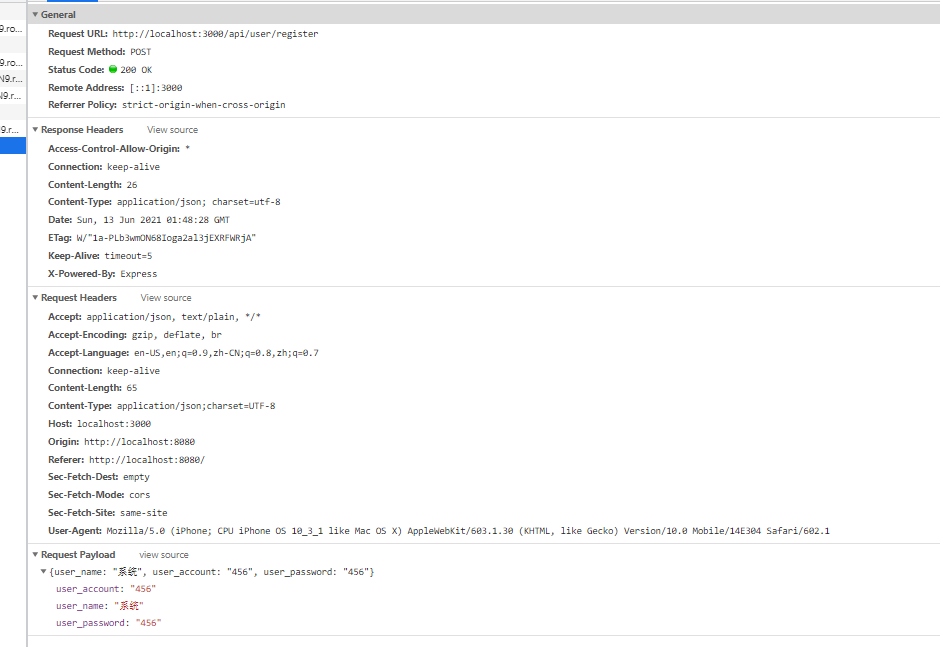




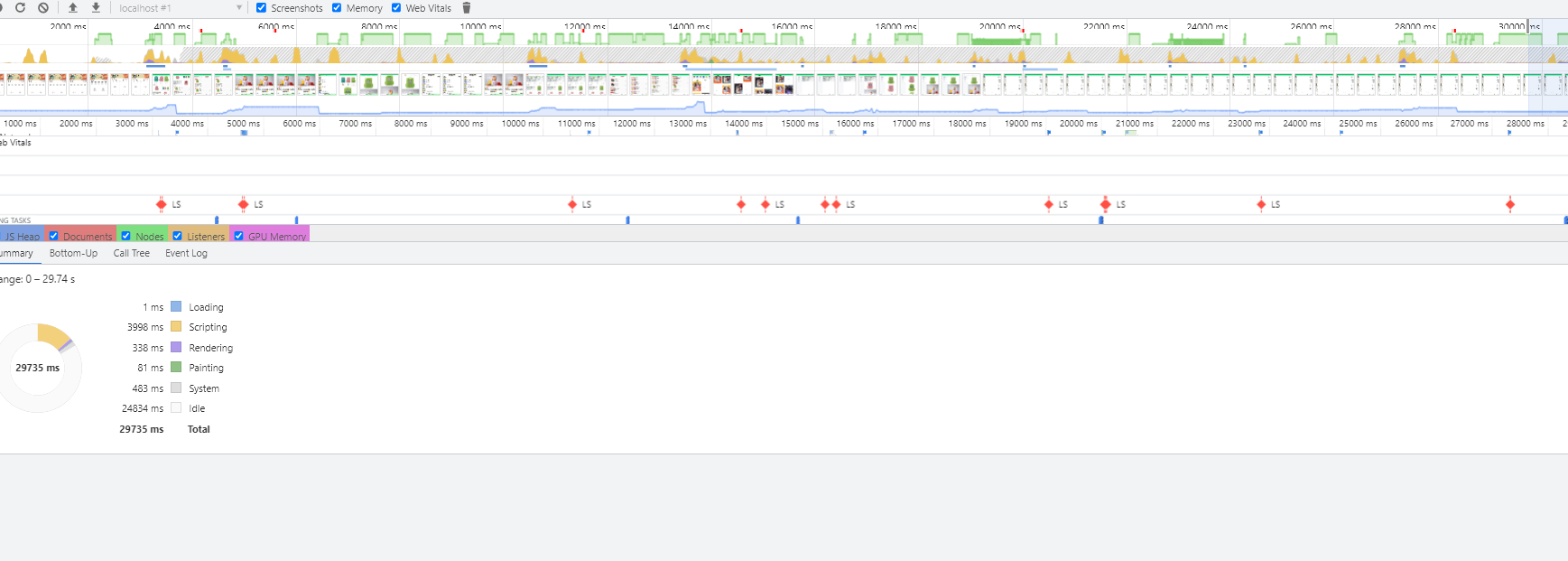




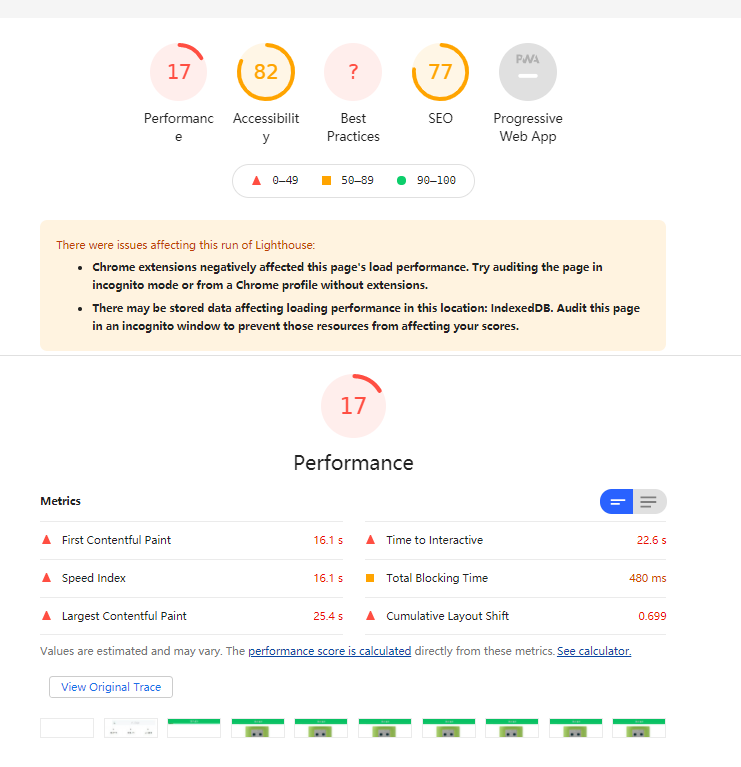


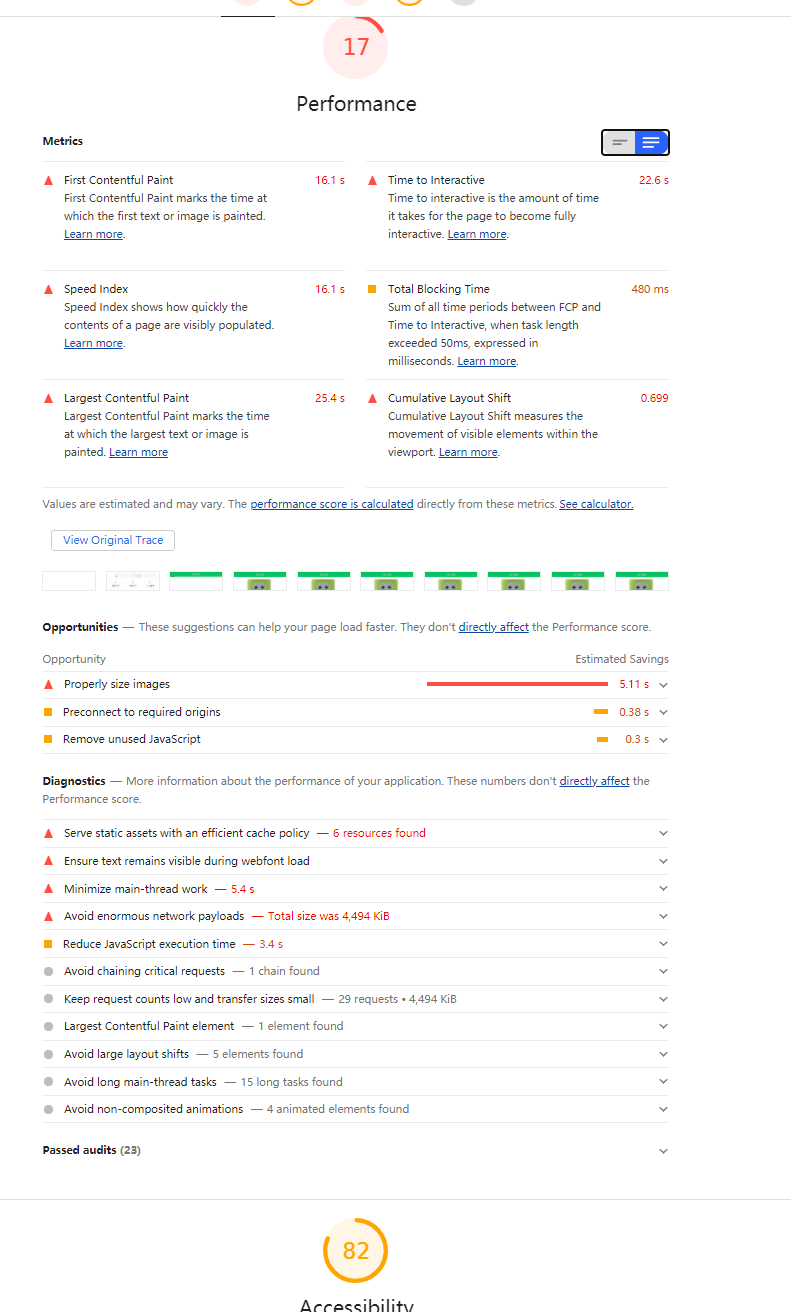


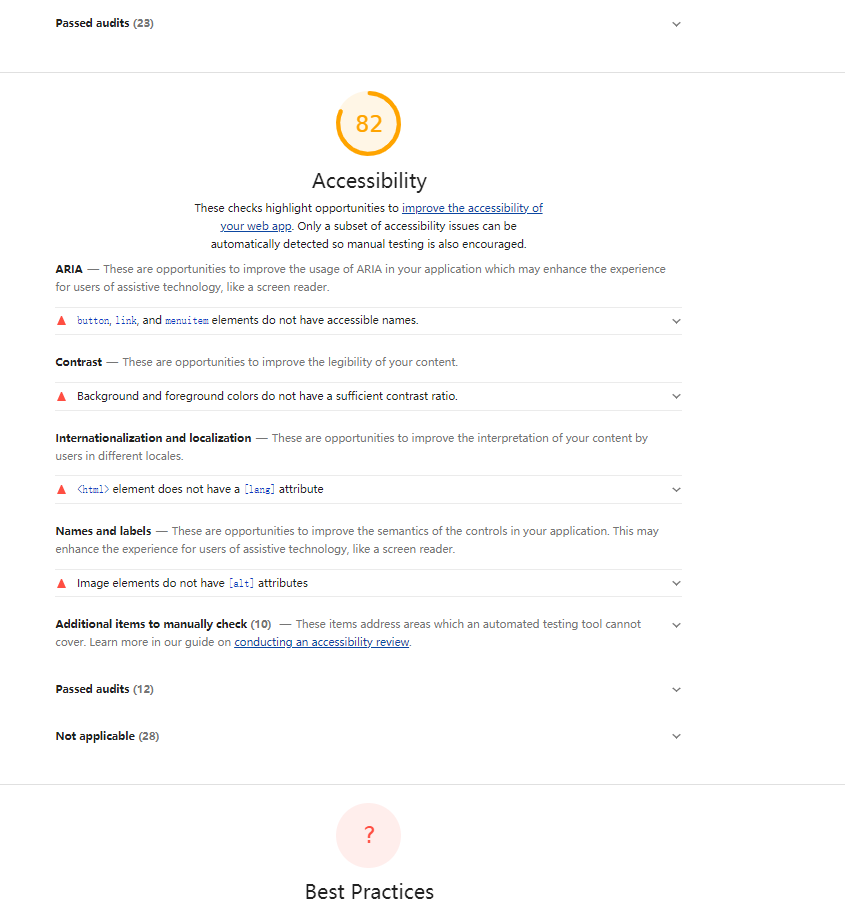
Performance性能分析

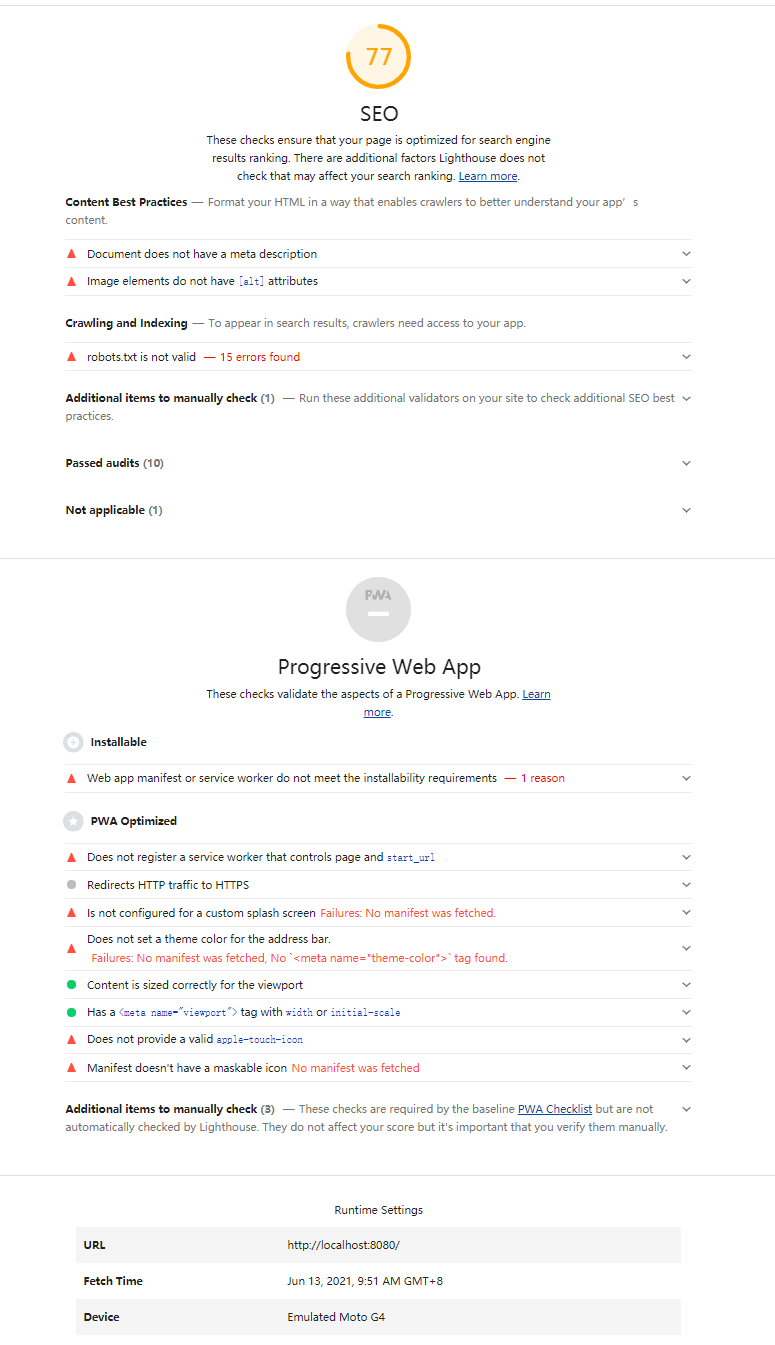


LightHouse性能分析









Express 后端部分

框架 采用Express Mysql

MySql 工具使用了navicat

文件目录：

├─.idea 这是编译器的配置信息

│ └─inspectionProfiles

├─bin 这是主要执行的点

├─controller 这里是对数据库的操作

├─dao 这里是连接数据库的池

├─middles 这里放着中间件

├─node\_modules 这是项目依赖

├─public 这是打包后的Vue（run build 后的）文件

│ └─static

│ ├─css

│ ├─fonts

│ └─js

│ ─index.html

├─routes 这里是路由层配置

└─services 这里是类似与orm，对sql进行的封装

Package.json中的项目依赖

{

"name": "shop\_system",

"version": "0.0.0",

"private": true,

"scripts": {

"start": "nodemon ./bin/www"

},

"dependencies": {

"cookie-parser": "~1.4.4",

"cores": "^0.8.5",

"debug": "~2.6.9",

"ejs": "~2.6.1",

"express": "~4.16.1",

"express-session": "^1.17.2",

"http-errors": "~1.6.3",

"mockjs": "^1.1.0",

"morgan": "~1.9.1",

"multer": "^1.4.2",

"mysql2": "^2.2.5",

"vue-puzzle-vcode": "^1.1.5"

},

"devDependencies": {

"cors": "^2.8.5",

"jsonwebtoken": "^8.5.1"

}

}

数据表分布

