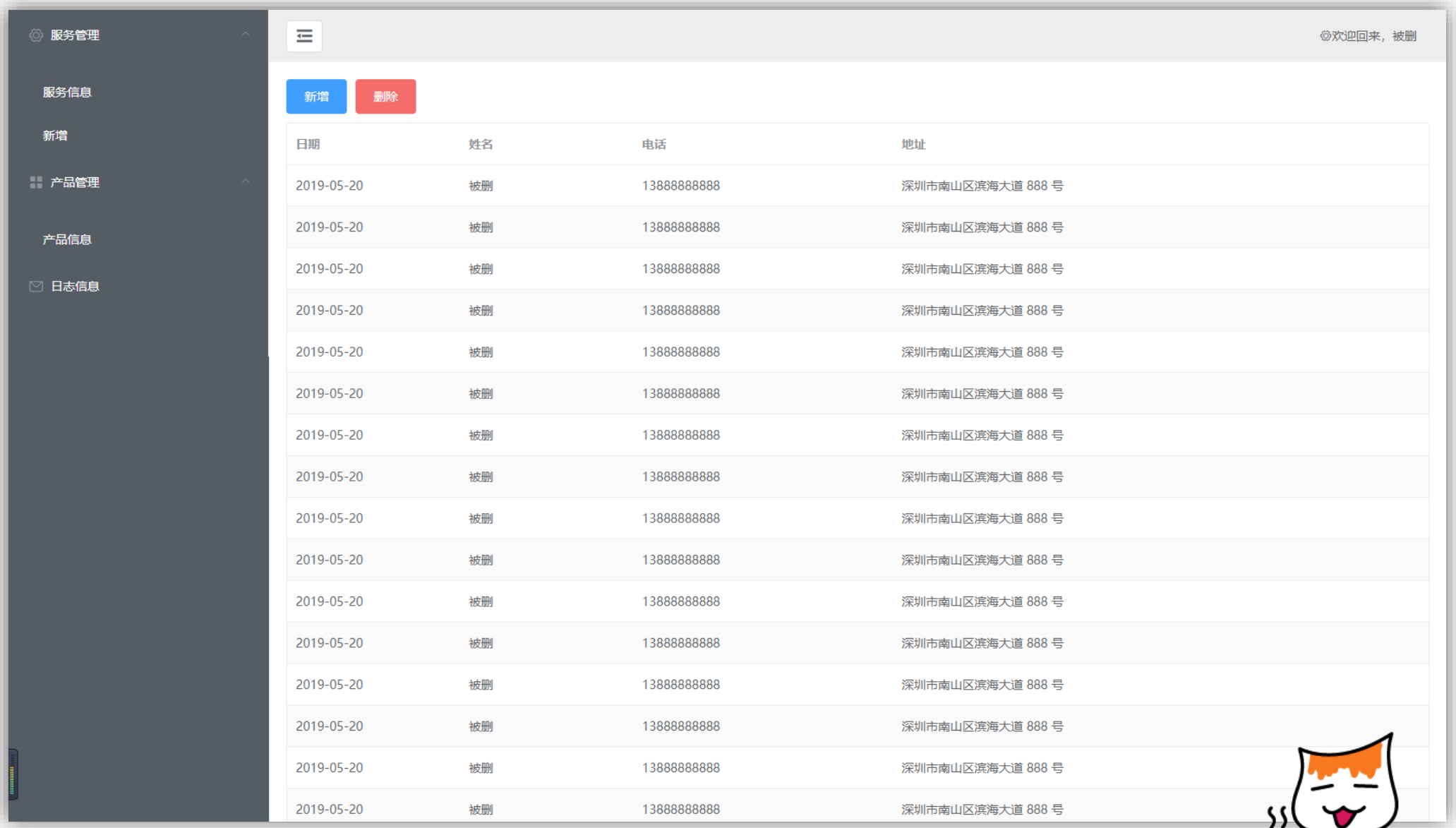
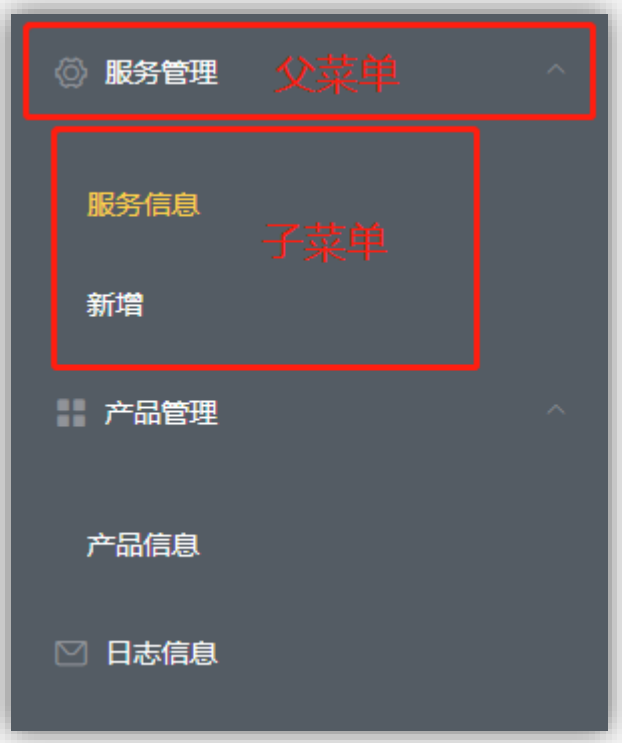
如何3天实现管理端（下）

@被删

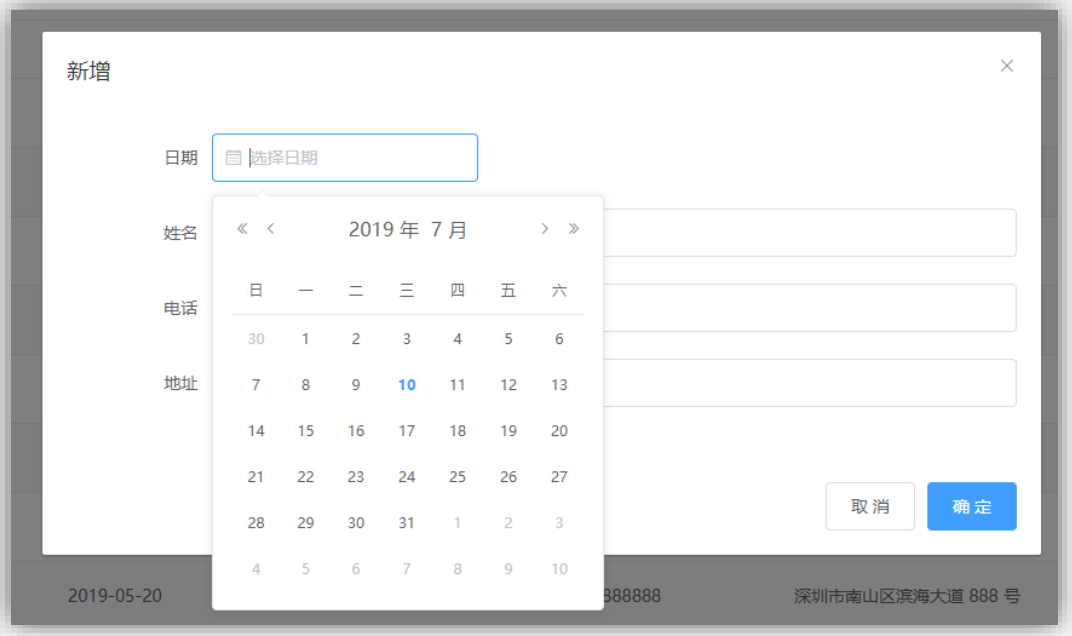


管理端基本上都是增删查改

各个组件的实现


服务管理
父菜单
服务信息
子菜单
新增
产品管理
产品信息
日志信息


新增
删除
日期
姓名
电话
地址
2019-05-20
被删
13888888888
深圳市南山区滨海大道 888 号
2019-05-20
被删
13888888888
深圳市南山区滨海大道 888 号
2019-05-20
被删
13888888888
深圳市南山区滨海大道 888 号


新增
日期
选择日期
姓名
电话
地址
2019 年 7 月
日 一 二 三 四 五 六
30 1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31 1 2 3
4 5 6 7 8 9 10
取 消
确 定
2019-05-20
深圳市南山区滨海大道 888 号

3 管理端路由设计

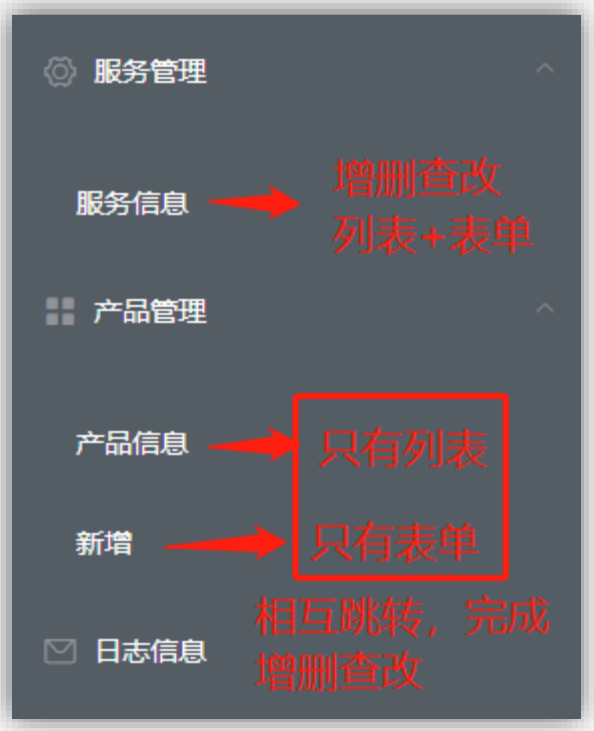
管理端页面类型

管理端页面基本上由以下类型组成：

序号	页面形式	页面能力
1	登录页	只有用户名和密码的输入
2	列表 + 表单	单页可以完成某类服务的增删查改
3	列表页	• 只有列表展示 • 提供查和删服务 • 需要配合 4 的表单页完成增和改
4	表单页	• 只有表单编辑内容 • 可提供新增、修改等能力给 3 使用

管理端路由设计

根据管理端的页面内容和形式，进行路由设计

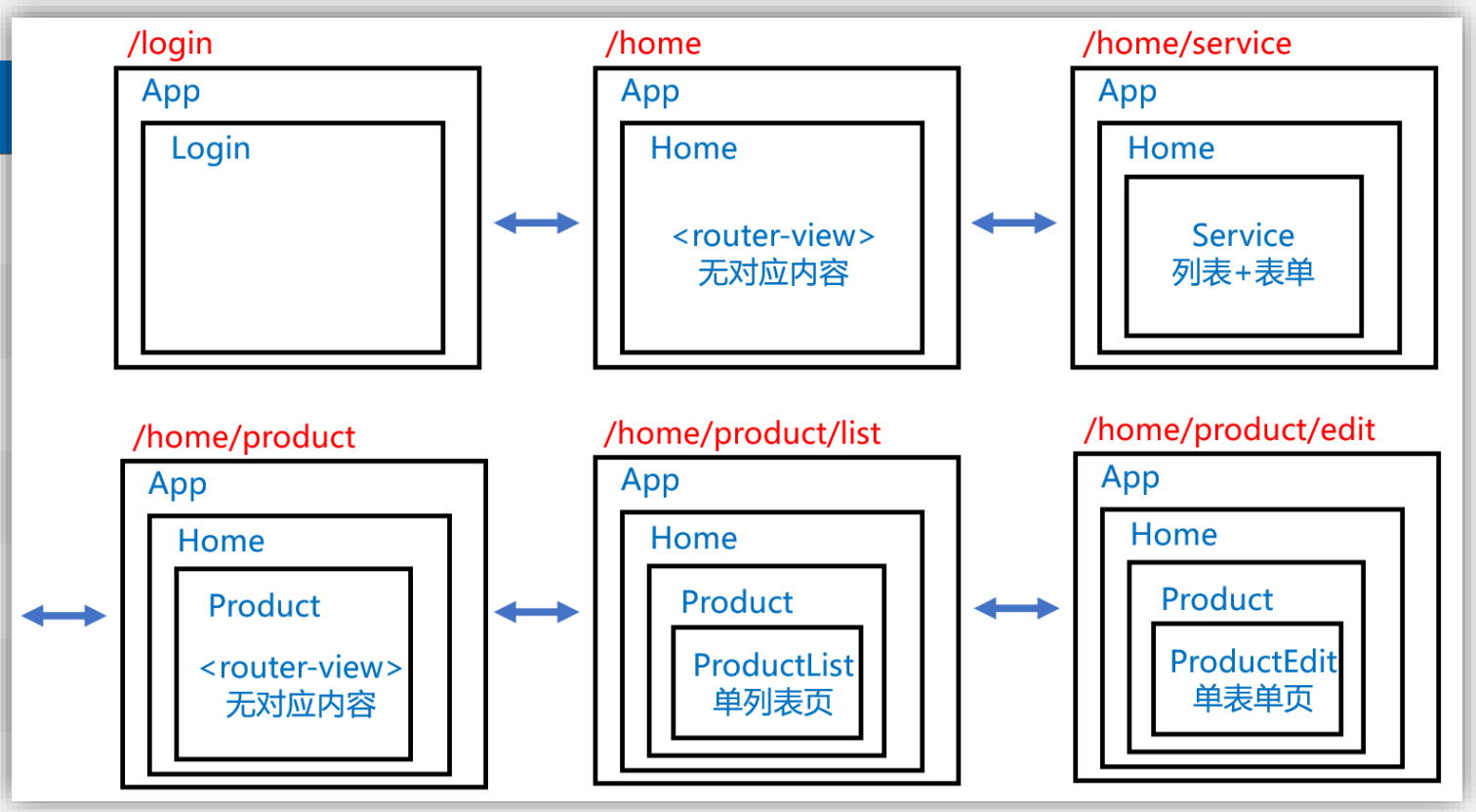


路由	页面内容	页面对应的Component	页面组成
/login	登录页	Login	• 表单 • 包括username和password
/home	应用首页	Home	• 左侧菜单<Menu> • 右侧路由内容<router-view>
/home/service	服务信息页	Service	• 为 Home 的子路由组件 • 包括列表和表单
/home/product	产品容器页	Product	• 为 Home 的子路由组件 • 包括<router-view>
/home/product/list	产品信息页	ProductList	• 为 Product 的子路由组件 • 包括列表
/home/product/edit	产品编辑页	ProductEdit	• 为 Product 的子路由组件 • 包括表单

管理端路由设计

根据管理端的页面内容和路由，进行组件嵌套

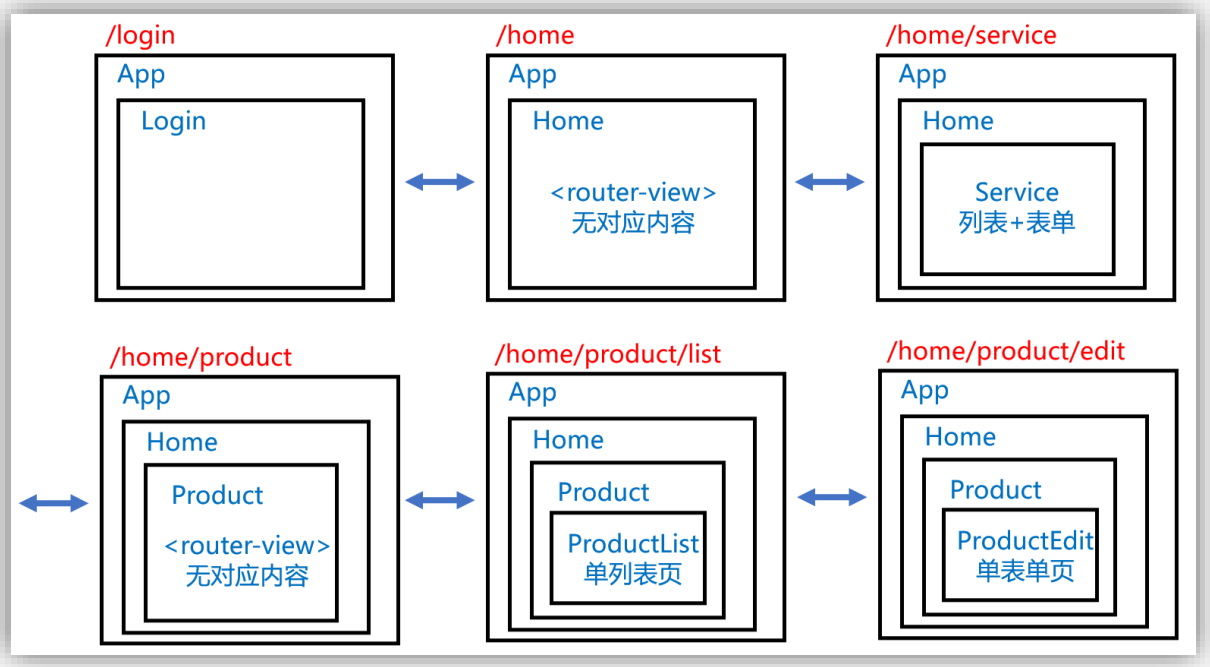
路由	页面内容	页面对应的Component
/login	登录页	Login
/home	应用首页	Home
/home/service	服务信息页	Service
/home/product	产品容器页	Product
/home/product/list	产品信息页	ProductList
/home/product/edit	产品编辑页	ProductEdit



4 路由具体实现

路由实现—路由配置

1. 整体路由配置




```
const routes = [
  {
    path: "/", // 父路由路径
    component: App, // 父路由组件，传入 vue component
    name: "App", // 路由名称
    // 设置子路由
    children: [
      {
        path: "login", // 子路由路径
        component: Login, // 子路由组件，会替换父组件中<router-view>中的内容
        name: "Login" // 路由名称
      },
      {
        // 应用首页
        path: "home",
        component: Home,
        name: "Home",
        children: [
          // 服务列表
          { path: "service", component: Service, name: "Service" },
          // 产品容器
          {
            path: "product",
            component: Product,
            name: "Product",
            children: [
              // 子路由内容
              // 产品列表
              { path: "list", component: ProductList, name: "ProductList" },
              // 产品新增
              { path: "add/0", component: ProductEdit, name: "ProductAdd" },
              // 产品编辑
              // 我们能看到，新增和编辑其实最终使用的是同一个组件，所以后面会有一些需要兼容处理的地
              // :id可匹配任意值，且可在组件中通过this.$route.params.id获取该值
              { path: "edit/:id", component: ProductEdit, name: "ProductEdit" }
            ]
          }
        ]
      }
    ]
  }
];
```

路由实现—路由实现

2. 路由加载和 <router-view> 嵌套

```
// 加载路由信息
const router = new VueRouter({
  // mode: 路由模式，'hash' | 'history'
  // routes: 传入路由配置信息，后面会讲怎么配置
  routes // （缩写）相当于 routes: routes
});
// 启动一个 Vue 应用
new Vue({
  el: "#app",
  router, // 传入路由能力
  render: h => h(App)
});
```

加载路由

```
<!-- 这里是最外层 /app 路由的组件，App.vue -->
<template>
  <!-- 使用 <router-view></router-view> 来嵌套路由 -->
  <router-view></router-view>
</template>

<!-- 这里是 /app/home 路由的组件，Home.vue -->
<!-- 这里采用了简写，省略了一些非关键内容，更多内容可以参考上一节 -->
<template>
  <el-container>
    <!-- 左侧菜单，Menu.vue -->
    <menu></menu>
    <!-- 右侧内容 -->
    <el-container>
      <!-- 上边的头部栏 -->
      <el-header></el-header>
      <!-- 子路由页面的内容 -->
      <router-view></router-view>
    </el-container>
  </el-container>
</template>
```

<router-view>

路由实现—路由实现

3. 菜单绑定路由

```
<el-menu-item-group>
  <!-- 使用 router-link 组件来导航. -->
  <!-- 通过传入 `to` 属性指定链接. -->
  <!-- <router-link> 默认会被渲染成一个 `<a>` 标签 -->
  <router-link
    tag="div"
    v-for="subMenu in menu.subMenus"
    :key="subMenu.index"
    :to="{name: subMenu.routerName}"
  >
    <el-menu-item :index="subMenu.index">{{subMenu.text}}</el-menu-item>
  </router-link>
</el-menu-item-group>
```

<router-link>

```
// 下面是 Vue 组件
export default {
  data() {
    return {
      menus, // menus: menus 的简写
      activeIndex: ""
    };
  },
  watch: {
    // 为了设置 el-menu 的 default-active 属性，需要获取到路由状态
    $route(to) {
      // 对路由变化作出响应...
      let activeIndex;
      // 找到匹配的 routerName
      this.menus.forEach(x => {
        if (x.routerName === to.name) {
          activeIndex = x.index;
        } else {
          const subMenuItem = x.subMenus.find(y => y.routerName === to.name);
          if (subMenuItem) {
            activeIndex = subMenuItem.index;
          }
        }
      });
      // 并将 activeIndex 设置为对应的 菜单 index
      if (activeIndex) {
        this.activeIndex = activeIndex;
      }
    }
  }
};
```

监控路由的变化，并根据路由情况调整绑定的激活 index

路由实现—路由跳转

4. 路由跳转

```
export default {
  // ...其他省略
  methods: {
    // 新增/修改一个数据
    updateTableItem(id = 0) {
      // 跳转到编辑页面，新增则传id为0，否则为编辑
      // 可以通过 this.$router 访问路由实例
      if (id !== 0) {
        // 传参 name 为路由名字，params 为我们定义的路由 path 的参数，变成 /edit/xxx
        // 还有另外一种传参方式 query，带查询参数，变成 /edit?id=xxx
        this.$router.push({ name: "ProductEdit", params: { id } });
      } else {
        this.$router.push({ name: "ProductAdd" });
      }
    }
  }
};
```

通过 $router 访问路由实例

路由实现—路由鉴权

5. 路由鉴权

```
import { setGlobalData } from "utils/globalData";

// 下面是 Vue 组件
export default {
  // ...其他省略
  methods: {
    // 提交新增/修改表单
    onSubmit() {
      // 校验表单，用户名和密码都必须填入
      // Element 表单校验规则配置，请查看https://element.eleme.cn/#/zh-CN/component/form
      this.$refs["form"].validate(valid => {
        if (valid) {
          // 校验通过
          // 设置用户名
          setGlobalData("username", this.form.username);
          // 跳转到里页
          this.$router.push({ name: "Home" });
        } else {
          // 校验失败
          return false;
        }
      });
    }
  }
};
```

登录页面登录

```
// main.js
import { getGlobalData } from "utils/globalData";

router.beforeEach((to, from, next) => {
  if (to.name !== "Login") {
    // 非 login 页面，检查是否登录
    // 这里简单前端模拟是否填写了用户名，真实环境需要走后台登录，缓存到本地
    if (!getGlobalData("username")) {
      next({ name: "Login" });
    }
  }
  // 其他情况正常执行
  next();
});
```

鉴权进入内页

深入理解Vue.js实战

作者：被删

第二部分 Vue的正确使用方式

第9章 思维转变与大型项目管理

第10章 如何正确地进行抽象

第11章 全局数据管理与 Vuex

第12章 实战：三天开发一个管理端

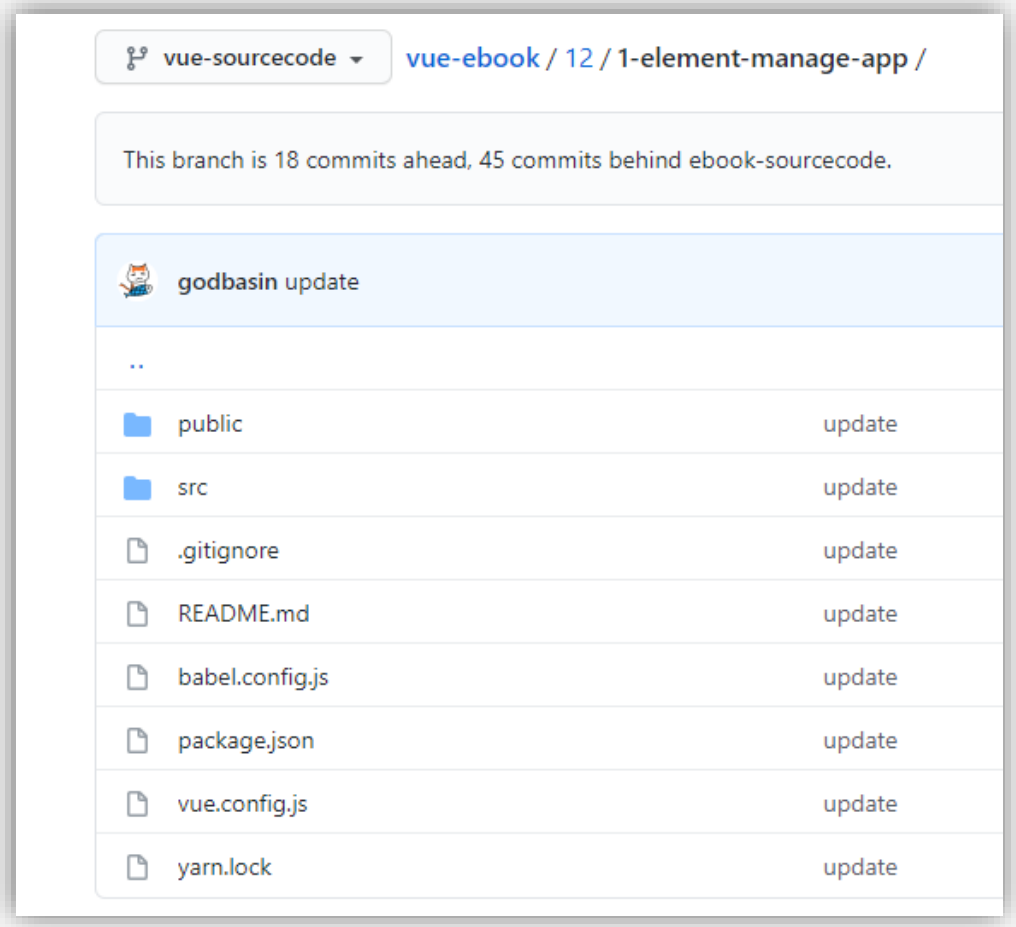
12.1 设计管理端功能

12.2 组件快速开发

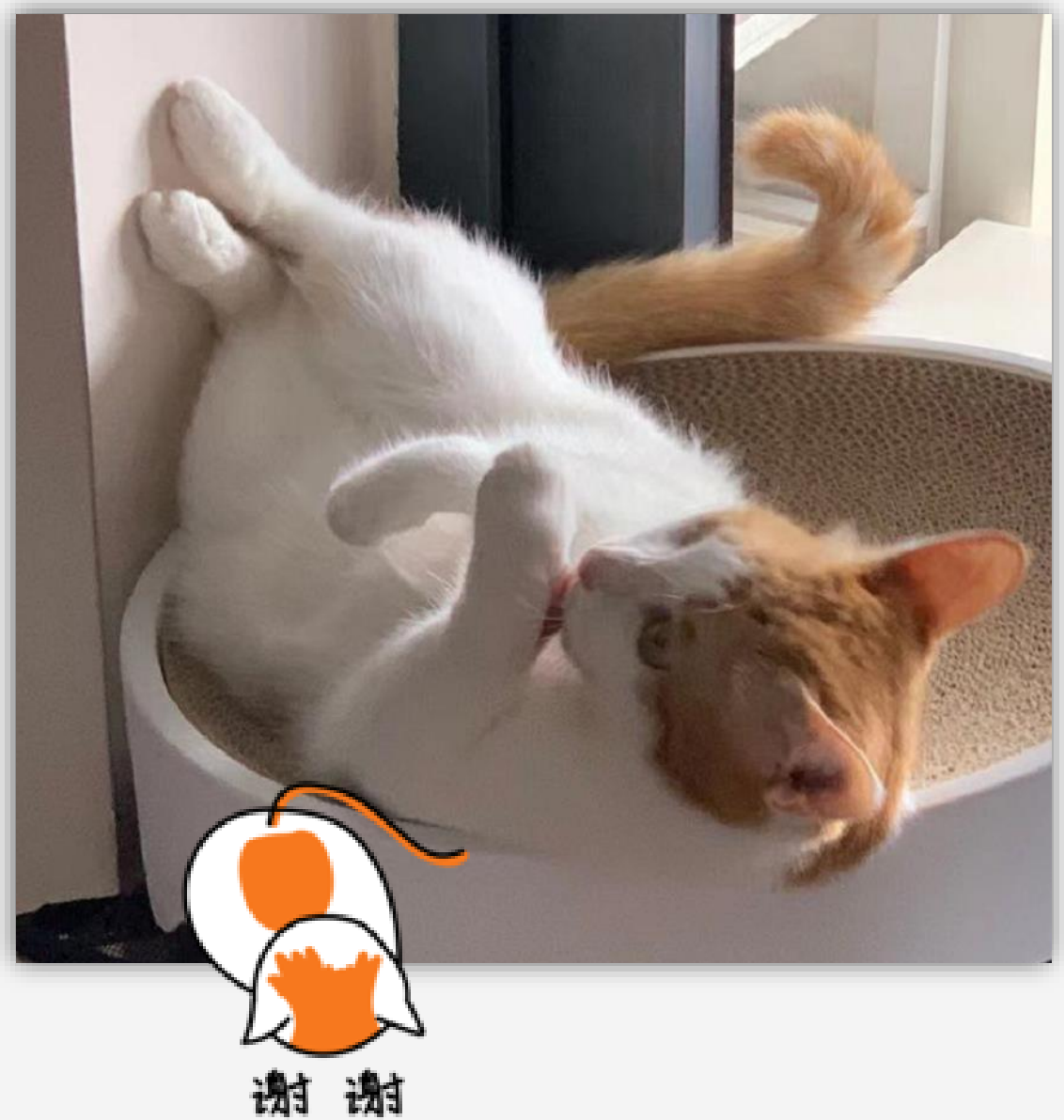
12.3 设计页面与路由

12.4 路由配置与开发

12.5 给路由添加鉴权



https://github.com/godbasin/vue-ebook/tree/vue-sourcecode



Github: godbasin
@被删