

第四节、图书借阅数据分析系统客户端设计与实现

4.1.hbuilder开发工具下载与安装



4.2.node服务器下载与安装

1.3.1.node.js服务器安装 访问网址:<http://nodejs.cn/>



4.3.vueCli安装

/**npm简介

node.js服务器安装 访问网址:<http://nodejs.cn/9.3.vueCli简介>

NPM是随同NodeJS一起安装的包管理工具，能解决NodeJS代码部署上的很多问题，常见的使用场景有以下几种：

允许用户从NPM服务器下载别人编写的第三方包到本地使用。

允许用户从NPM服务器下载并安装别人编写的命令行程序到本地使用。

允许用户将自己编写的包或命令程序上传到NPM服务器供别人使用。

由于新版的nodejs已经集成了npm，所以之前npm也一并安装好了。

同样可以通过输入 "npm -v" 来测试是否成功安装。命令如下，出现版本提示表示安装成功*/

/**

Vue CLI 是一个基于 Vue.js 进行快速开发的完整系统，通过 @vue/cli 快速创建前端工程

配置阿里加速地址： npm install -g cnpm --registry=https://registry.npm.taobao.org

安装命令 npm install -g @vue/cli

查看版本命令vue --version

使用webpack创建项目

步骤1 进入要创建项目的盘符或者文件夹输入如下命令： vue init webpack 工程名称

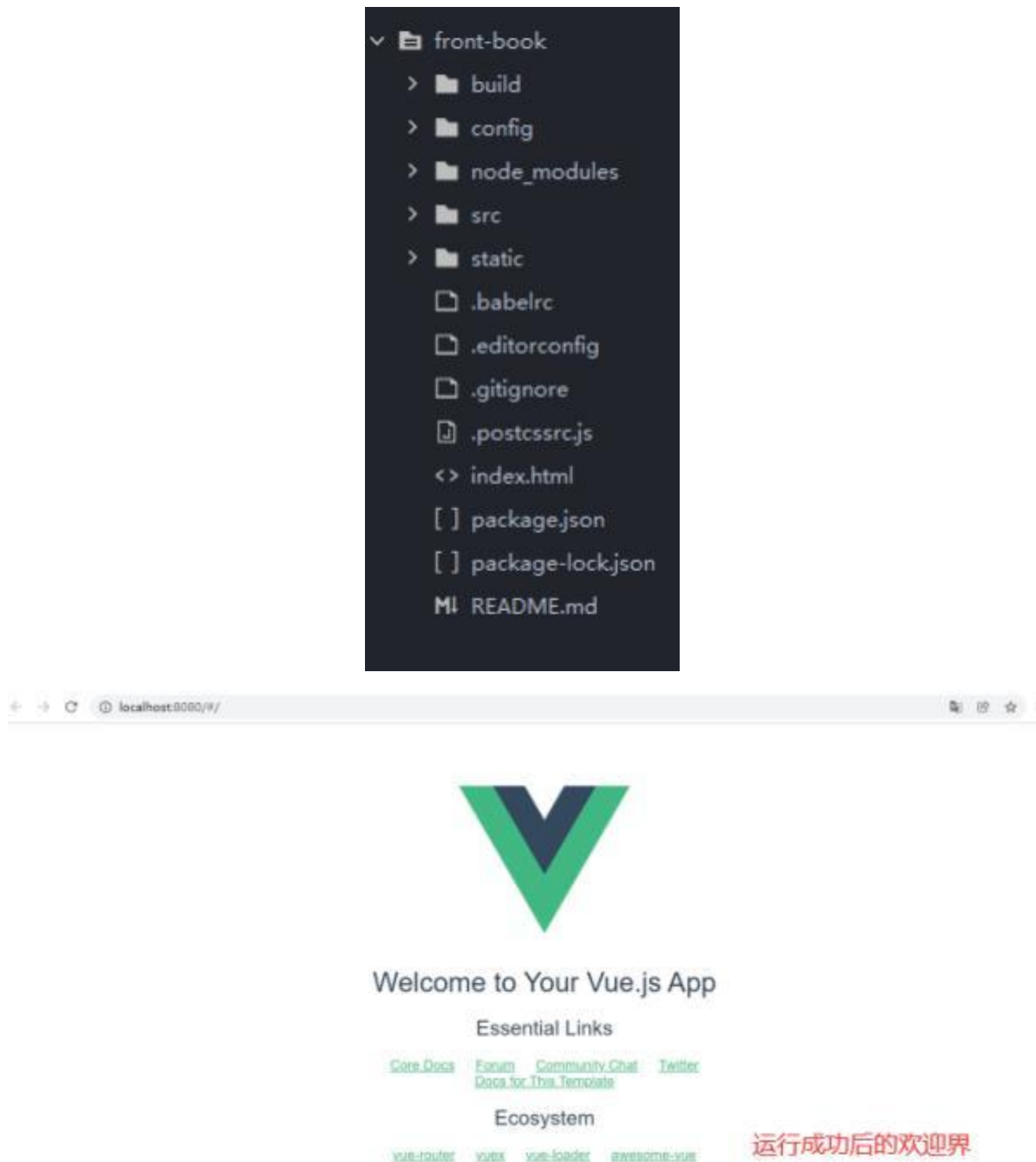
*/

```
Microsoft Windows [版本 10.0.19042.1415]
(c) Microsoft Corporation。保留所有权利。

F:\HBuilderProjects>vue init webpack front-vrms          使用命令创建工程，工程名称为front-vrms。回车

'git'  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 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 
```

4.4客户端工程目录结构



第五节、图书借阅数据分析系统工程前端架构搭建

5.1.Element概述

Element，一套为开发者、设计师和产品经理准备的基于 Vue 2.0 的桌面端组件库，官网：<https://element.eleme.cn/#/zh-CN>

5.2.Element开发环境构建

```
//开发环境安装进入工程目录运行如下命令：npm i element-ui -S
//在main.js中引入如下命令
import ElementUI from 'element-ui';
import 'element-ui/lib/theme-chalk/index.css';
vue.use(ElementUI);
```

5.3.axios概述

```
//axios是vue框架提倡使用的ajax库
//axios安装 进入工程目录使用 npm install axios -S命令安装
//main.js中引用
import axios from 'axios'
vue.prototype.$http=axios
```

5.4.样式引入

```
//在工程静态资源中编写main.css
html,
body,
#app {
margin: 0px;
height: 100%;
padding: 0px;
}
/
/在main.js中引入
import './assets/css/main.css'
```

5.5.局部组件Main.vue创建

```
<template>
<el-container>
<el-header>疫苗预约管理系统</el-header>
<el-container>
<el-aside width="200px">导航菜单</el-aside>
<el-main>主体</el-main>
</el-container>
</el-container>
</template>
<script>
export default {
data() {
return {}
}
}
```

```

</script>
<style scoped>
.el-header {
background-color: #2B4B6B;
display: flex;
justify-content: space-between;
padding-left: 10px;
align-items: center;
color: white;
font-size: 18px;
} .
.el-aside {
color: white;
background-color: #2B4B6B;
} .
.el-main {
background-color: aliceblue;
} .
.el-container {
height: 100%;}
</style>

```



5.6.导航菜单设计与实现

```

//在路由(router)文件夹中的index.js中编写如下代码
import Vue from 'vue'
import Router from 'vue-router'
import Main from '../components/Main.vue'
import Category from '../components/Category.vue'
import Book from '../components/Book.vue'
import Logs from '../components/Logs.vue'
import Borrow from '../components/Borrow.vue'
import Users from '../components/Users.vue'
import Welcome from '../components/welcome.vue'
Vue.use(Router)

export default new Router({
  routes: [
    {

```

```

    path: '/',
    name: 'Main',
    component: Main,
    redirect : '/welcome',
    children:[
      {
        path: '/welcome',
        name: welcome,
        component: welcome
      },
      {
        path: '/category',
        name: 'Category',
        component: Category,
      },{
        path: '/book',
        name: 'book',
        component: Book,
      },{
        path: '/borrow',
        name: 'Borrow',
        component: Borrow,
      },{
        path: '/users',
        name: 'Users',
        component: Users,
      },{
        path: '/logs',
        name: 'Logs',
        component: Logs,
      }
    ]
  }
]
})

```

```

<template>
  <el-container>
    <el-header>图书借阅管理系统</el-header>
    <el-container>
      <el-aside width="200px">
        <el-menu router background-color="#2B4B6B" text-color="#fff"
          active-text-color="#409EFF" unique-opened>
          <el-submenu index="3">
            <template slot="title">
              <i class="el-icon-s-tools"></i>
              <span>馆藏类别管理</span>
            </template>
            <el-menu-item index="/category">
              <i class="el-icon-menu"></i>
              <span slot="title">类别信息管理</span>
            </el-menu-item>=
            <el-menu-item index="/book">
              <i class="el-icon-reading"></i>
              <span slot="title">馆藏图书管理</span>
            </el-menu-item>
          </el-submenu>
        </el-menu>
      </el-aside>
    </el-container>
  </el-container>
</template>

```

```

        </el-menu-item>
      </el-submenu>
      <el-submenu index="2">
        <template slot="title">
          <i class="el-icon-s-cooperation"></i>
          <span>借阅信息管理</span>
        </template>
        <el-menu-item index="/borrow">
          <i class="el-icon-files"></i>
          <span slot="title">借阅信息管理</span>
        </el-menu-item>
      </el-submenu>
      <el-submenu index="1">
        <template slot="title">
          <i class="el-icon-s-platform"></i>
          <span>系统信息管理</span>
        </template>
        <el-menu-item index="/users">
          <i class="el-icon-user"></i>
          <span slot="title">用户信息管理</span>
        </el-menu-item>
        <el-menu-item index="/logs">
          <i class="el-icon-document"></i>
          <span slot="title">日志信息管理</span>
        </el-menu-item>
      </el-submenu>
    </el-menu>
  </el-aside>
  <el-main>
    <!-- 路由占位符 -->
    <router-view/>
  </el-main>
</el-container>
</el-container>
</template>

<script>
  export default {
    data() {
      return {

      }
    }
  }
</script>

<style scoped>
  .el-header {
    background-color: #2B4B6B;
    display: flex;
    justify-content: space-between;
    padding-left: 10px;
    align-items: center;
    color: white;
    font-size: 18px;
  }

```

```

}

.el-aside {
  color: white;
  background-color: #2B4B6B;
}

.el-main {
  background-color: aliceblue;
}

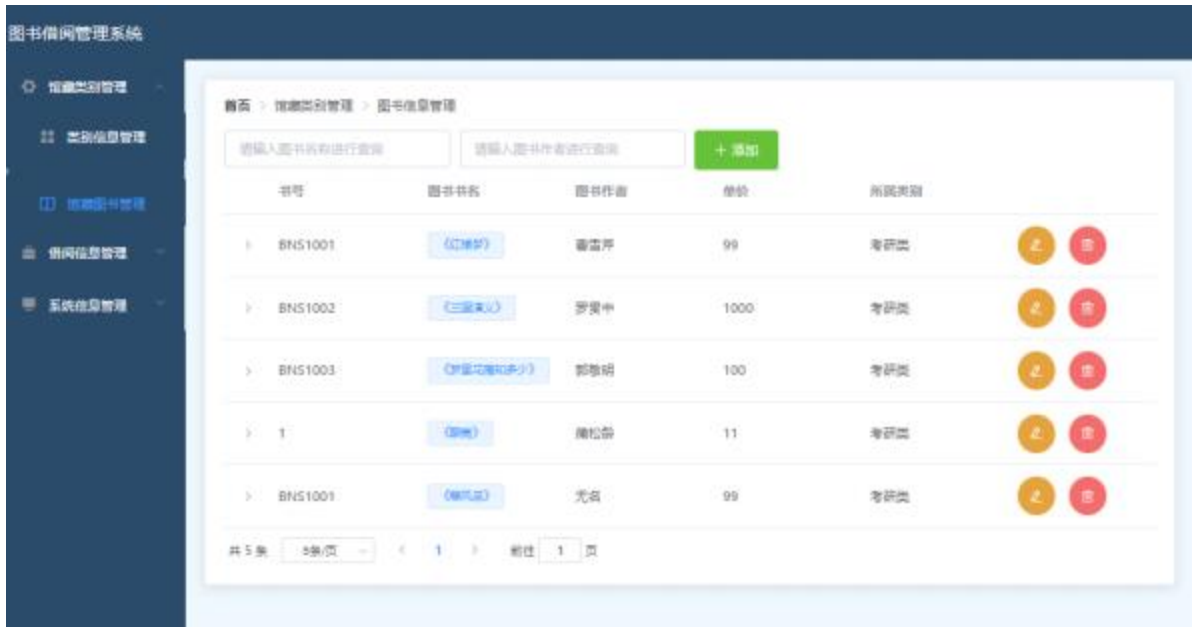
.el-container {
  height: 100%;
}
</style>

```



第六节、图书借阅数据分析系统前端各个UI模块设计与实现

6.1.馆藏图书模块UI设计与实现



```
<template>
  <!-- 卡片组件 -->
  <el-card>
    <!-- 面包屑组件 -->
    <el-breadcrumb separator-class="el-icon-arrow-right">
      <el-breadcrumb-item :to="{ path: '/welcome' }">首页</el-breadcrumb-item>
      <el-breadcrumb-item>馆藏类别管理</el-breadcrumb-item>
      <el-breadcrumb-item>图书信息管理</el-breadcrumb-item>
    </el-breadcrumb>
    <el-row :gutter="10">
      <el-col :span="6">
        <el-input clearable @change="listPage()" v-model="book.bname"
placeholder="请输入图书名称进行查询"></el-input>
      </el-col>
      <el-col :span="6">
        <el-input clearable @change="listPage()" v-model="book.author"
placeholder="请输入图书作者进行查询"></el-input>
      </el-col>
      <el-col :span="12">
        <el-button icon="el-icon-plus" @click="openAddDialog" type="success">添加
</el-button>
      </el-col>
    </el-row>
    <!-- 表格组件 -->
    <el-table :data="tableData">
      <el-table-column type="expand" prop="ddesc">
        <template slot-scope="scope">
          <el-tag>简介: {{scope.row.ddesc}}</el-tag>
        </template>
      </el-table-column>
      <el-table-column prop="bnum" label="书号">
      </el-table-column>
      <el-table-column prop="bname" label="图书书名">
        <template slot-scope="scope">
          <el-popover trigger="hover" placement="top">
            <p>图书简介: {{ scope.row.ddesc }}</p>
            <p>库存数量: {{ scope.row.quantity }}</p>
            <div slot="reference" class="name-wrapper">
```

```

        <el-tag size="medium">{{ scope.row.bname }}</el-tag>
      </div>
    </el-popover>
  </template>
</el-table-column>
<el-table-column prop="author" label="图书作者">
</el-table-column>

<el-table-column prop="price" label="单价">
</el-table-column>

<el-table-column prop="category" label="所属类别">
  <template slot-scope="scope">
    {{ scope.row.category.catename }}
  </template>
</el-table-column>
<el-table-column label="操作">
  <!-- 作用域插槽 -->
  <template slot-scope="scope">
    <el-tooltip content="点击修改" placement="bottom">
      <el-button type="warning" @click="openEditDialog(scope.row)"
icon="el-icon-edit" circle></el-button>
    </el-tooltip>
    <el-tooltip content="点击删除" placement="bottom">
      <el-button type="danger" @click="remove(scope.row)" icon="el-icon-
delete" circle></el-button>
    </el-tooltip>
  </template>
</el-table-column>
</el-table>

<!-- 分页组件定义 -->
<el-pagination @size-change="handleSizeChange" @current-
change="handleCurrentChange" :current-page="pager.page"
:page-sizes="[5, 10, 15, 20]" :page-size="pager.size" layout="total,
sizes, prev, pager, next, jumper"
:total="pager.total">
</el-pagination>

<!-- 添加对话框 -->
<el-dialog title="图书类别添加" :show-close="false" :close-on-click-
modal="false" :visible.sync="addDialogFlag">
  <el-form :model="book" :rules="rules" ref="book">
    <el-form-item label="类别名称" prop="catename">
      <el-input v-model="book.catename" placeholder="请输入类别名称"></el-
input>
    </el-form-item>
  </el-form>
  <div slot="footer" class="dialog-footer">
    <el-button @click="addDialogFlag= !addDialogFlag">取 消</el-button>
    <el-button type="primary" @click="save">保存</el-button>
  </div>
</el-dialog>

<!-- 修改对话框 -->

```

```

<el-dialog title="图书类别修改" :show-close="false" :close-on-click-
modal="false" :visible.sync="editDialogFlag">

  <div slot="footer" class="dialog-footer">
    <el-button @click="editDialogFlag= !editDialogFlag">取 消</el-button>
    <el-button type="primary" @click="update">保存</el-button>
  </div>
</el-dialog>

</el-card>

</template>

<script>
export default {
  data() {
    return {
      rules: {
        catename: [{
          required: true,
          message: '请输入类别名称 ',
          trigger: 'blur'
        }]
      }, //定义表单验证规则
      editDialogFlag: false, //控制修改对话框显示
      book: { //封装数据
        bid: 0,
        bname: '',
        author: '',
        price: '',
        quantity: '',
        ddesc: '',
        cateid: 0
      },
      addDialogFlag: false, //控制对话框是否显示
      cateName: '', //条件查询
      tableData: [],
      pager: { //分页封装对象
        page: 1,
        size: 5,
        total: 0
      }
    }
  },
  //该属性是vue对象中的成员变量，作用编写用户自己定的方法
  methods: {
    openAddDialog() {
      this.addDialogFlag = !this.addDialogFlag; //将打开添加对话框标记改成true
    },
    openEditDialog(row) {
      this.category = row;
      console.info(this.category)
      this.editDialogFlag = !this.editDialogFlag;
    }
  }
}

```

```

},
update() {
  this.$refs['category'].validate(valid => {
    if (valid) {
      this.$http.post('http://127.0.0.1/category/update', this.category)
        .then(res => {
          if (res.data.code == 200) {
            this.$message({
              message: res.data.message,
              type: 'success'
            });
            this.listPage();
            this.editDialogFlag = !this.editDialogFlag;
          }
        }).catch(error => {
          this.$message.error('错了哦，网络异常');
        })
    }
  })
},
save() {
  // let parmas = new URLSearchParams();
  // parmas.append("catename", this.category.catename)
  this.$refs['category'].validate(valid => {
    if (valid) {
      this.$http.post('http://127.0.0.1/category/add', this.category)
        .then(res => {
          if (res.data.code == 200) {
            this.$message({
              message: res.data.message,
              type: 'success'
            });
            this.listPage();
            this.addDialogFlag = !this.addDialogFlag;
          }
        }).catch(error => {
          this.$message.error('错了哦，网络异常');
        })
    }
  })
},
remove(row) {
  this.$http.post('http://127.0.0.1/category/remove?cateid=' + row.cateid)
    .then(res => {
      if (res.data.code == 200) {
        this.$message({
          message: res.data.message,
          type: 'success'
        });
        this.listPage();
      }
    })
}

```

```

    }).catch(error => {
      this.$message.error('错了哦，网络异常');
    })
  },

  handleCurrentChange(val) {
    this.pager.page = val;
    this.listPage();
  },
  handleSizeChange(val) {
    this.pager.size = val;
    this.listPage(); //重新调用查询方法
  },
  //用户定义的前端请求方法
  listPage() {
    this.$http.get('http://127.0.0.1/book/listPage', {
      params: {
        page: this.pager.page,
        size: this.pager.size,
        bname: this.book.bname,
        author: this.book.author
      }
    }).then(res => {

      this.tableData = res.data.data.data;
      this.pager.total = res.data.data.total
    }).catch(error => {
      this.$message.error('错了哦，网络异常');
    })
  }

},
//vue声明周期函数
mounted() {
  this.listPage();
}
}
</script>

<style>
.el-breadcrumb {
  padding-bottom: 20px;
}

.el-pagination {
  padding-top: 10px;
}
</style>

```

6.2.图书类别模块UI设计与实现



```
<template>
  <!-- 卡片组件 -->
  <el-card>
    <!-- 面包屑组件 -->
    <el-breadcrumb separator-class="el-icon-arrow-right">
      <el-breadcrumb-item :to="{ path: '/welcome' }">首页</el-breadcrumb-item>
      <el-breadcrumb-item>馆藏类别管理</el-breadcrumb-item>
      <el-breadcrumb-item>类别信息管理</el-breadcrumb-item>
    </el-breadcrumb>
    <el-row :gutter="10">
      <el-col :span="8">
        <el-input clearable @change="listPage()" v-model="cateName"
placeholder="请输入图书类别名称查询"></el-input>
      </el-col>
      <el-col :span="12">
        <el-button icon="el-icon-plus" @click="openAddDialog" type="success">添加
</el-button>
      </el-col>
    </el-row>
    <!-- 表格组件 -->
    <el-table :data="tableData" style="width: 100%">
      <el-table-column prop="catename" label="类别名称" width="180">
</el-table-column>
      <el-table-column prop="createtime" label="创建时间">
</el-table-column>
      <el-table-column prop="modifytime" label="修改时间">
</el-table-column>
      <el-table-column label="操作">
        <!-- 作用域插槽 -->
        <template slot-scope="scope">
          <el-tooltip content="点击修改" placement="bottom">
            <el-button type="warning" @click="openEditDialog(scope.row)"
icon="el-icon-edit" circle></el-button>
          </el-tooltip>
          <el-tooltip content="点击删除" placement="bottom">
            <el-button type="danger" @click="openDeleteDialog(scope.row)"
icon="el-icon-delete" circle></el-button>
          </el-tooltip>
        </template>
      </el-table-column>
    </el-table>
  </el-card>
</template>
```

```

        <el-button type="danger" @click="remove(scope.row)" icon="el-icon-
delete" circle></el-button>
      </el-tooltip>
    </template>
  </el-table-column>
</el-table>

<!-- 分页组件定义 -->
<el-pagination @size-change="handleSizeChange" @current-
change="handleCurrentChange" :current-page="pager.page"
:page-sizes="[5, 10, 15, 20]" :page-size="pager.size" layout="total,
sizes, prev, pager, next, jumper"
:total="pager.total">
</el-pagination>

<!-- 添加对话框 -->
<el-dialog title="图书类别添加" :show-close="false" :close-on-click-
modal="false" :visible.sync="addDialogFlag">
  <el-form :model="category" :rules="rules" ref="category">
    <el-form-item label="类别名称" prop="catename">
      <el-input v-model="category.catename" placeholder="请输入类别名称"></el-
input>
    </el-form-item>
  </el-form>
  <div slot="footer" class="dialog-footer">
    <el-button @click="addDialogFlag= !addDialogFlag">取 消</el-button>
    <el-button type="primary" @click="save">保存</el-button>
  </div>
</el-dialog>

<!-- 修改对话框 -->
<el-dialog title="图书类别修改" :show-close="false" :close-on-click-
modal="false" :visible.sync="editDialogFlag">
  <el-form :model="category" :rules="rules" ref="category">
    <el-form-item label="类别名称" prop="catename">
      <el-input v-model="category.catename" placeholder="请输入类别名称"></el-
input>
    </el-form-item>
  </el-form>
  <div slot="footer" class="dialog-footer">
    <el-button @click="editDialogFlag= !editDialogFlag">取 消</el-button>
    <el-button type="primary" @click="update">保存</el-button>
  </div>
</el-dialog>

</el-card>

</template>

<script>
export default {
  data() {
    return {
      rules: {

```

```

        catename: [{
            required: true,
            message: '请输入类别名称',
            trigger: 'blur'
        }]
    }, //定义表单验证规则
    editDialogFlag: false, //控制修改对话框显示
    category: { //封装数据
        cateid: 0,
        catename: ''
    },
    addDialogFlag: false, //控制对话框是否显示
    cateName: '', //条件查询
    tableData: [],
    pager: { //分页封装对象
        page: 1,
        size: 5,
        total: 0
    }
}
},
//该属性是vue对象中的成员变量，作用编写用户自己定的方法
methods: {
    openAddDialog() {
        this.addDialogFlag = !this.addDialogFlag; //将打开添加对话框标记改成true
    },
    openEditDialog(row) {
        this.category = row;
        console.info(this.category)
        this.editDialogFlag = !this.editDialogFlag;
    },
    update() {
        this.$refs['category'].validate(valid => {
            if (valid) {
                this.$http.post('http://127.0.0.1/category/update', this.category)
                    .then(res => {
                        if (res.data.code == 200) {
                            this.$message({
                                message: res.data.message,
                                type: 'success'
                            });
                            this.listPage();
                            this.editDialogFlag = !this.editDialogFlag;
                        }
                    })
                    .catch(error => {
                        this.$message.error('错了哦，网络异常');
                    })
            }
        })
    },
    save() {
        // let parmas = new URLSearchParams();
        // parmas.append("catename", this.category.catename)
        this.$refs['category'].validate(valid => {

```



```

        if (valid) {
            this.$http.post('http://127.0.0.1/category/add', this.category)
                .then(res => {
                    if (res.data.code == 200) {
                        this.$message({
                            message: res.data.message,
                            type: 'success'
                        });
                        this.listPage();
                        this.addDialogFlag = !this.addDialogFlag;
                    }
                }).catch(error => {
                    this.$message.error('错了哦，网络异常');
                })
        }
    })

},
remove(row) {
    this.$http.post('http://127.0.0.1/category/remove?cateid=' + row.cateid)
        .then(res => {
            if (res.data.code == 200) {
                this.$message({
                    message: res.data.message,
                    type: 'success'
                });
                this.listPage();
            }

        }).catch(error => {
            this.$message.error('错了哦，网络异常');
        })
},

handleCurrentChange(val) {
    this.pager.page = val;
    this.listPage();
},
handleSizeChange(val) {
    this.pager.size = val;
    this.listPage(); //重新调用查询方法
},

//用户定义的前端请求方法
listPage() {
    this.$http.get('http://127.0.0.1/category/listPage', {
        params: {
            page: this.pager.page,
            size: this.pager.size,
            cateName: this.cateName
        }
    }).then(res => {

        this.tableData = res.data.data.data;
    })
}

```

```

        this.pager.total = res.data.data.count
    }).catch(error => {
        this.$message.error('错了哦，网络异常');
    })
}

},
//vue声明周期函数
mounted() {
    this.listPage();
}
}
</script>

<style>
.el-breadcrumb {
    padding-bottom: 20px;
}

.el-pagination {
    padding-top: 10px;
}
</style>

```

6.3.借阅模块UI设计与实现



```

<template>
  <!-- 卡片组件 -->
  <el-card>
    <!-- 面包屑组件 -->
    <el-breadcrumb separator-class="el-icon-arrow-right">
      <el-breadcrumb-item :to="{ path: '/welcome' }">首页</el-breadcrumb-item>
      <el-breadcrumb-item>借阅信息管理</el-breadcrumb-item>
      <el-breadcrumb-item>借阅信息管理</el-breadcrumb-item>
    </el-breadcrumb>
    <el-row :gutter="10">
      <el-col :span="8">
        <el-input clearable @change="listPage()" v-model="borrow.book.bname"
placeholder="请输入借阅图书称查询"></el-input>

```

```

        </el-col>
        <el-col :span="12">
            <el-button icon="el-icon-plus" @click="openAddDialog" type="success">借阅
        </el-button>
        </el-col>
    </el-row>
    <!-- 表格组件 -->
    <el-table :data="tableData">
        <el-table-column prop="book" label="图书名称">
            <template slot-scope="scope">
                {{scope.row.book.bname}}
            </template>
        </el-table-column>
        <el-table-column prop="users" label="借阅人">
            <template slot-scope="scope">
                {{scope.row.users.uname}}
            </template>
        </el-table-column>
        <el-table-column prop="otime" label="借出时间">
        </el-table-column>
        <el-table-column prop="itime" label="归还时间">
        </el-table-column>
        <el-table-column prop="quantity" label="数量">
        </el-table-column>
        <el-table-column prop="quantity" label="状态">
            <template slot-scope="scope">
                <el-tag type="success" v-show="scope.row.stat==1">已归还</el-tag>
                <el-tag type="danger" v-show="scope.row.stat==0">已借出</el-tag>
            </template>
        </el-table-column>
        <el-table-column label="操作">
            <!-- 作用域插槽 -->
            <template slot-scope="scope">
                <el-button size="mini" type="danger" v-show="scope.row.stat==0">归还
            </el-button>
            </template>
        </el-table-column>
    </el-table>
    <!-- 分页组件定义 -->
    <el-pagination @size-change="handleSizeChange" @current-
change="handleCurrentChange" :current-page="pager.page"
:page-sizes="[5, 10, 15, 20]" :page-size="pager.size" layout="total,
sizes, prev, pager, next, jumper"
:total="pager.total">
    </el-pagination>
</el-card>

</template>

<script>
export default {
  data() {
    return {

```

```

rules: {
  catename: [{
    required: true,
    message: '请输入类别名称',
    trigger: 'blur'
  }]
}, //定义表单验证规则
editDialogFlag: false, //控制修改对话框显示
category: { //封装数据
  cateid: 0,
  catename: ''
},
borrow: {
  boid: 0,
  otime: '',
  itime: '',
  quantity: 0,
  book: {
    bid: 0,
    bname: ''
  },
  users: {
    uid: 0,
    uname: ''
  }
},
addDialogFlag: false, //控制对话框是否显示
cateName: '', //条件查询
tableData: [],
pager: { //分页封装对象
  page: 1,
  size: 5,
  total: 0
}
},
//该属性是vue对象中的成员变量，作用编写用户自己定的方法
methods: {
  openAddDialog() {
    this.addDialogFlag = !this.addDialogFlag; //将打开添加对话框标记改成true
  },
  openEditDialog(row) {
    this.category = row;
    console.info(this.category)
    this.editDialogFlag = !this.editDialogFlag;
  },

  handleCurrentChange(val) {
    this.pager.page = val;
    this.listPage();
  },
  handleSizeChange(val) {
    this.pager.size = val;
    this.listPage(); //重新调用查询方法
  }
}

```

```

    },
    //用户定义的前端请求方法
    listPage() {
      this.$http.get('http://127.0.0.1/borrow/listPage', {
        params: {
          page: this.pager.page,
          size: this.pager.size,
          bname: this.borrow.book.bname
        }
      }).then(res => {

        this.tableData = res.data.data.data;
        this.pager.total = res.data.data.count
      }).catch(error => {
        this.$message.error('错了哦，网络异常');
      })
    }

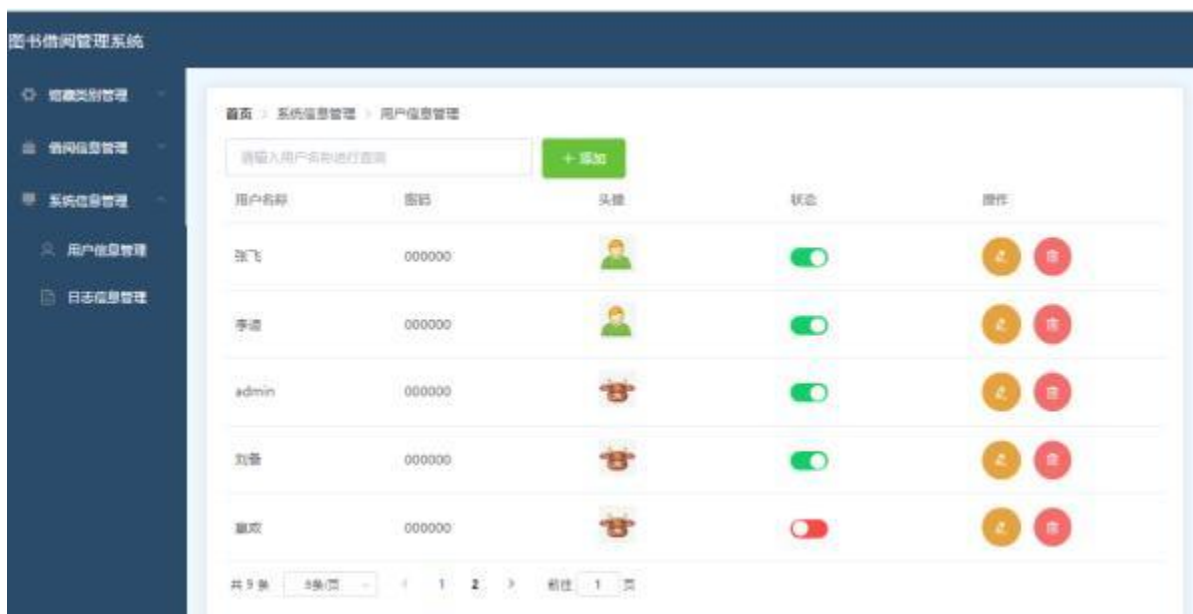
  },
  //vue声明周期函数
  mounted() {
    this.listPage();
  }
}
</script>

<style>
.el-breadcrumb {
  padding-bottom: 20px;
}

.el-pagination {
  padding-top: 10px;
}
</style>

```

6.4.用户模块UI设计与实现



```

<template>
  <!-- 卡片组件 -->
  <el-card>
    <!-- 面包屑组件 -->
    <el-breadcrumb separator-class="el-icon-arrow-right">
      <el-breadcrumb-item :to="{ path: '/welcome' }">首页</el-breadcrumb-item>
      <el-breadcrumb-item>系统信息管理</el-breadcrumb-item>
      <el-breadcrumb-item>用户信息管理</el-breadcrumb-item>
    </el-breadcrumb>
    <el-row :gutter="10">
      <el-col :span="8">
        <el-input clearable @change="listPage()" v-model="uname"
          placeholder="请输入用户名称进行查询"></el-input>
      </el-col>
      <el-col :span="12">
        <el-button icon="el-icon-plus" @click="openAddDialog" type="success">添加
      </el-button>
      </el-col>
    </el-row>
    <!-- 表格组件 -->
    <el-table :data="tableData" style="width: 100%">
      <el-table-column prop="uname" label="用户名称" width="180">
      </el-table-column>
      <el-table-column prop="password" label="密码">
      </el-table-column>
      <el-table-column prop="images" label="头像">
        <template slot-scope="scope">
          
        </template>
      </el-table-column>
      <el-table-column prop="stats" label="状态">
        <template slot-scope="scope">
          <el-switch
            v-model="scope.row.stats"
            :active-value="0"
            :inactive-value="1"
            active-color="#13ce66"
            inactive-color="#ff4949" @change="changStats(scope.row)">
          </el-switch>
        </template>
      </el-table-column>
      <el-table-column label="操作">
        <!-- 作用域插槽 -->
        <template slot-scope="scope">
          <el-tooltip content="点击修改" placement="bottom">
            <el-button type="warning" @click="openEditDialog(scope.row)"
              icon="el-icon-edit" circle></el-button>
          </el-tooltip>
          <el-tooltip content="点击删除" placement="bottom">
            <el-button type="danger" @click="remove(scope.row)" icon="el-icon-
              delete" circle></el-button>
          </el-tooltip>
        </template>
      </el-table-column>

```

```

</el-table>
<!-- 分页组件定义 -->
<el-pagination @size-change="handleSizeChange" @current-
change="handleCurrentChange" :current-page="pager.page"
:page-sizes="[5, 10, 15, 20]" :page-size="pager.size" layout="total,
sizes, prev, pager, next, jumper"
:total="pager.total">
</el-pagination>

<!-- 添加对话框 -->
<el-dialog title="用户添加" :show-close="false" :close-on-click-modal="false"
:visible.sync="addDialogFlag">
  <el-form :model="users" :rules="rules" ref="users">
    <el-form-item label="用户名称" prop="catename">
      <el-input v-model="users.uname" placeholder="请输入用户名称"></el-input>
    </el-form-item>
    <el-form-item label="用户密码" prop="catename">
      <el-input v-model="users.password" placeholder="请输入用户密码"></el-
input>
    </el-form-item>
    <el-form-item label="用户头像" prop="catename">
      <el-input v-model="users.images" placeholder="请输入用户头像"></el-input>
    </el-form-item>
  </el-form>
  <div slot="footer" class="dialog-footer">
    <el-button @click="addDialogFlag= !addDialogFlag">取 消</el-button>
    <el-button type="primary" @click="save">保存</el-button>
  </div>
</el-dialog>

<!-- 修改对话框 -->
<el-dialog title="用户修改" :show-close="false" :close-on-click-modal="false"
:visible.sync="editDialogFlag">
  <el-form :model="users" :rules="rules" ref="users">
    <el-form-item label="用户名称" prop="catename">
      <el-input v-model="users.uname" placeholder="请输入用户名称"></el-input>
    </el-form-item>
    <el-form-item label="用户密码" prop="catename">
      <el-input v-model="users.password" placeholder="请输入用户密码"></el-
input>
    </el-form-item>
    <el-form-item label="用户头像" prop="catename">
      <el-input v-model="users.images" placeholder="请输入用户头像"></el-input>
    </el-form-item>
  </el-form>
  <div slot="footer" class="dialog-footer">
    <el-button @click="editDialogFlag= !editDialogFlag">取 消</el-button>
    <el-button type="primary" @click="update">保存</el-button>
  </div>
</el-dialog>

</el-card>

```

```

</template>

<script>
  export default {
    data() {
      return {
        uname: '',
        rules: {
          uname: [{
            required: true,
            message: '请输入用户名称 ',
            trigger: 'blur'
          }],
          password: [{
            required: true,
            message: '请输入密码 ',
            trigger: 'blur'
          }],
          images: [{
            required: true,
            message: '请输入头像连接地址 ',
            trigger: 'blur'
          }]
        }, //定义表单验证规则
        editDialogFlag: false, //控制修改对话框显示
        users: { //封装数据
          uid: 0,
          uname: '',
          images: '',
          password: ''
        },
        addDialogFlag: false, //控制对话框是否显示
        tableData: [],
        pager: { //分页封装对象
          page: 1,
          size: 5,
          total: 0
        }
      }
    },
    //该属性是vue对象中的成员变量，作用编写用户自己定的方法
    methods: {
      changStats(row){ //改变用户状态
        this.users=row;

        this.$http.post('http://127.0.0.1/users/changStats', this.users).then(res=>{
          if (res.data.code == 200) {
            this.$message({
              message: res.data.message,
              type: 'success'
            });
            this.listPage();
          }
        })
      }
    }
  }

```



```

},
openAddDialog() {
  this.addDialogFlag = !this.addDialogFlag; //将打开添加对话框标记改成true
},
openEditDialog(row) {
  this.users = row;
  this.editDialogFlag = !this.editDialogFlag;
},
update() {
  this.$refs['users'].validate(valid => {
    if (valid) {
      this.$http.post('http://127.0.0.1/users/update', this.users)
        .then(res => {
          if (res.data.code == 200) {
            this.$message({
              message: res.data.message,
              type: 'success'
            });
            this.listPage();
            this.editDialogFlag = !this.editDialogFlag;
          }
        })
        .catch(error => {
          this.$message.error('错了哦，网络异常');
        })
    }
  })
},
save() {
  this.$refs['users'].validate(valid => {
    if (valid) {
      this.$http.post('http://127.0.0.1/users/save', this.users)
        .then(res => {
          if (res.data.code == 200) {
            this.$message({
              message: res.data.message,
              type: 'success'
            });
            this.listPage();
            this.addDialogFlag = !this.addDialogFlag;
          }
        })
        .catch(error => {
          this.$message.error('错了哦，网络异常');
        })
    }
  })
},
remove(row) {
  this.$http.post('http://127.0.0.1/users/remove?uid=' + row.uid)
    .then(res => {
      if (res.data.code == 200) {
        this.$message({
          message: res.data.message,

```

```

        type: 'success'
      });
      this.listPage();
    }

  }).catch(error => {
    this.$message.error('错了哦，网络异常');
  })
},

handleCurrentChange(val) {
  this.pager.page = val;
  this.listPage();
},

handleSizeChange(val) {
  this.pager.size = val;
  this.listPage(); //重新调用查询方法
},

//用户定义的前端请求方法
listPage() {
  this.$http.get('http://127.0.0.1/users/listPage', {
    params: {
      page: this.pager.page,
      size: this.pager.size,
      uname: this.uname
    }
  }).then(res => {

    this.tableData = res.data.data.data;
    this.pager.total = res.data.data.total
  }).catch(error => {
    this.$message.error('错了哦，网络异常');
  })
}

},

//vue声明周期函数
mounted() {
  this.listPage();
}
}
</script>

<style>
.el-breadcrumb {
  padding-bottom: 20px;
}

.el-pagination {
  padding-top: 10px;
}
</style>

```

6.5.日志模块UI设计与实现



```
<template>
  <!-- 卡片组件 -->
  <el-card>
    <!-- 面包屑组件 -->
    <el-breadcrumb separator-class="el-icon-arrow-right">
      <el-breadcrumb-item :to="{ path: '/welcome' }">首页</el-breadcrumb-item>
      <el-breadcrumb-item>系统信息管理</el-breadcrumb-item>
      <el-breadcrumb-item>日志信息管理</el-breadcrumb-item>
    </el-breadcrumb>

    <!-- 表格组件 -->
    <el-table :data="tableData" >
      <el-table-column type="expand">
        <template slot-scope="scope">
          <el-tag type="danger">操作方法:{{scope.row.methods}} {{scope.row.ddesc}}
        </el-tag>
      </el-table-column>
      <el-table-column prop="opername" label="操作人" width="180">
      </el-table-column>
      <el-table-column prop="opertime" label="操作时间">
      </el-table-column>
      <el-table-column prop="ip" label="操作IP">
      </el-table-column>
    </el-table>

    <!-- 分页组件定义 -->
    <el-pagination @size-change="handleSizeChange" @current-
change="handleCurrentChange" :current-page="pager.page"
      :page-sizes="[5, 10, 15, 20]" :page-size="pager.size" layout="total,
sizes, prev, pager, next, jumper"
      :total="pager.total">
    </el-pagination>
  </el-card>
</template>
```

```

<script>
export default {
  data() {
    return {
      rules: {
        catename: [{
          required: true,
          message: '请输入类别名称',
          trigger: 'blur'
        }]
      }, //定义表单验证规则
      editDialogFlag: false, //控制修改对话框显示
      category: { //封装数据
        cateid: 0,
        catename: ''
      },
      addDialogFlag: false, //控制对话框是否显示
      cateName: '', //条件查询
      tableData: [],
      pager: { //分页封装对象
        page: 1,
        size: 5,
        total: 0
      }
    }
  },
  //该属性是vue对象中的成员变量，作用编写用户自己定的方法
  methods: {

    handleCurrentChange(val) {
      this.pager.page = val;
      this.listPage();
    },
    handleSizeChange(val) {
      this.pager.size = val;
      this.listPage(); //重新调用查询方法
    },
    //用户定义的前端请求方法
    listPage() {
      this.$http.get('http://127.0.0.1/logs/listPage', {
        params: {
          page: this.pager.page,
          size: this.pager.size,
        }
      }).then(res => {

        this.tableData = res.data.data.data;
        this.pager.total = res.data.data.count
      }).catch(error => {
        this.$message.error('错了哦，网络异常');
      })
    }
  }
}

```

```
    },  
    //vue声明周期函数  
    mounted() {  
      this.listPage();  
    }  
  }  
</script>  
  
<style>  
  .el-breadcrumb {  
    padding-bottom: 20px;  
  }  
  
  .el-pagination {  
    padding-top: 10px;  
  }  
</style>
```