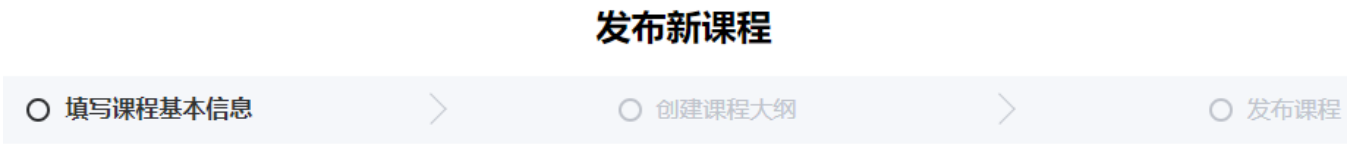
硅谷课堂第七天-点播管理模块（一）

一、后台管理系统-点播管理模块

1、点播管理模块需求

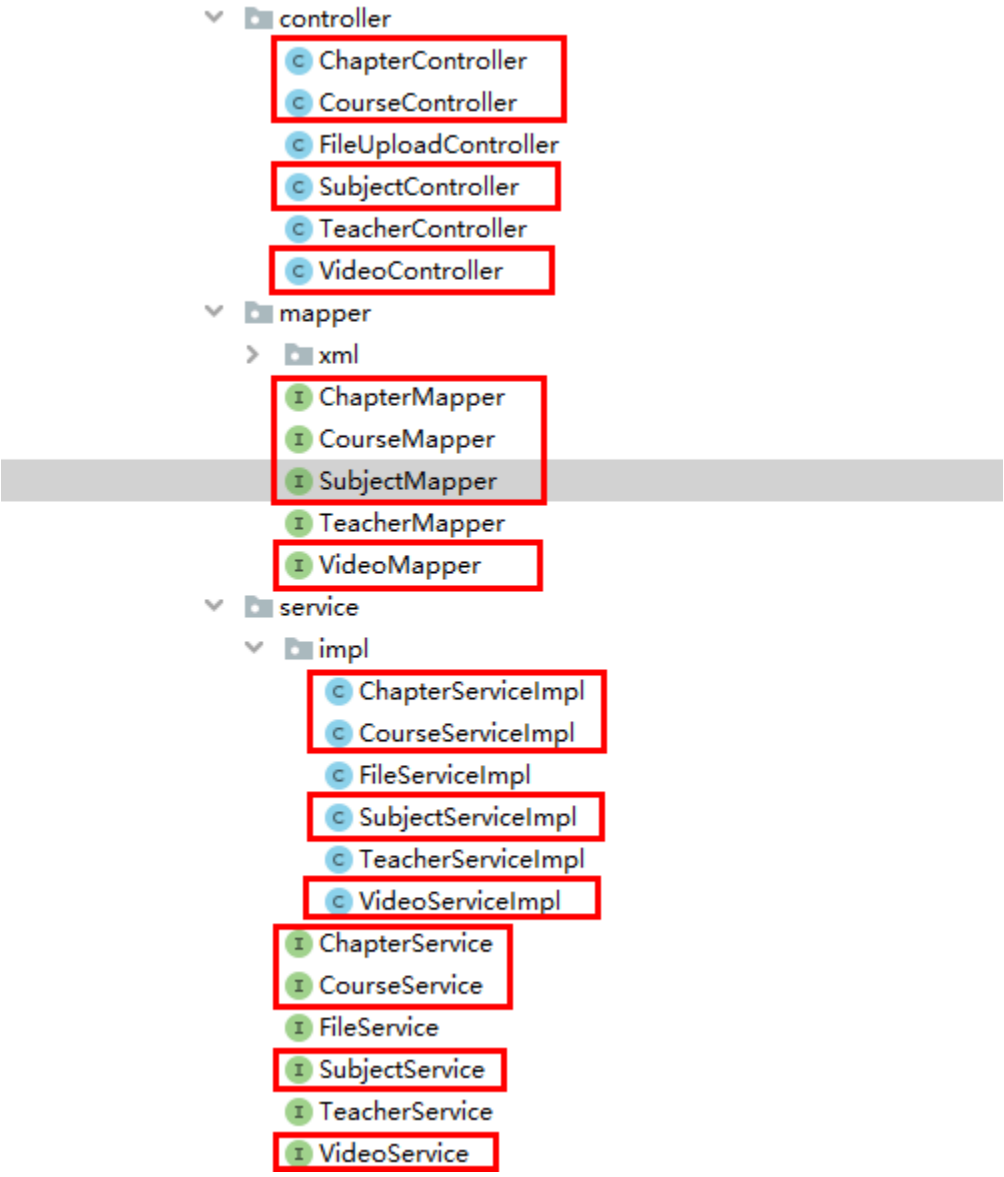
添加点播课程，包含课程基本信息，课程章节，课程小结和最终发布



1.1、创建课程相关表

2、环境搭建

2.1、生成相关代码



3、功能实现-课程列表

实现分页条件查询点播课程功能

3.1、开发课程列表接口

编写 CourseController

```
@Api(tags = "课程管理接口")
@RestController
@RequestMapping(value="/admin/vod/course")
public class CourseController {

    @Autowired
    private CourseService courseService;

    @ApiOperation(value = "获取分页列表")
    @GetMapping("{page}/{limit}")
    public Result index(
            @ApiParam(name = "page", value = "当前页码", required = true)
            @PathVariable Long page,
            @ApiParam(name = "limit", value = "每页记录数", required = true)
            @PathVariable Long limit,
            @ApiParam(name = "courseVo", value = "查询对象", required = false)
                    CourseQueryVo courseQueryVo) {
        Page<Course> pageParam = new Page<>(page, limit);
        Map<String,Object> map = courseService.findPage(pageParam, courseQueryVo);
        return Result.ok(map);
    }
}
```

编写 CourseService

```
public interface CourseService extends IService<Course> {
    //课程列表
    Map<String,Object> findPage(Page<Course> pageParam, CourseQueryVo courseQueryVo);
}
```

编写 CourseServiceImpl

```java
@Service
public class CourseServiceImpl extends ServiceImpl<CourseMapper, Course> implements CourseService

    @Autowired
    private TeacherService teacherService;

    @Autowired
    private SubjectService subjectService;

    //课程列表
    @Override
    public Map<String,Object> findPage(Page<Course> pageParam, CourseQueryVo courseQueryVo) {
        //获取条件值
        String title = courseQueryVo.getTitle();//名称
        Long subjectId = courseQueryVo.getSubjectId();//二级分类
        Long subjectParentId = courseQueryVo.getSubjectParentId();//一级分类
        Long teacherId = courseQueryVo.getTeacherId();//讲师
        //封装条件
        QueryWrapper<Course> wrapper = new QueryWrapper<>();
        if(!StringUtils.isEmpty(title)) {
            wrapper.like("title",title);
        }
        if(!StringUtils.isEmpty(subjectId)) {
            wrapper.eq("subject_id",subjectId);
        }
        if(!StringUtils.isEmpty(subjectParentId)) {
            wrapper.eq("subject_parent_id",subjectParentId);
        }
        if(!StringUtils.isEmpty(teacherId)) {
            wrapper.eq("teacher_id",teacherId);
        }
        //调用方法查询
        Page<Course> pages = baseMapper.selectPage(pageParam, wrapper);
        long totalCount = pages.getTotal();//总记录数
        long totalPage = pages.getPages();//总页数
        long currentPage = pages.getCurrent();//当前页
        long size = pages.getSize();//每页记录数
```

```
        //每页数据集合
        List<Course> records = pages.getRecords();
        //遍历封装讲师和分类名称
        records.stream().forEach(item -> {
            this.getTeacherOrSubjectName(item);
        });
        //封装返回数据
        Map<String,Object> map = new HashMap<>();
        map.put("totalCount",totalCount);
        map.put("totalPage",totalPage);
        map.put("records",records);
        return map;
    }

    //获取讲师和分类名称
    private Course getTeacherOrSubjectName(Course course) {
        //查询讲师名称
        Teacher teacher = teacherService.getById(course.getTeacherId());
        if(teacher != null) {
            course.getParam().put("teacherName",teacher.getName());
        }
    //查询分类名称
        Subject subjectOne = subjectService.getById(course.getSubjectParentId());
        if(subjectOne != null) {
            course.getParam().put("subjectParentTitle",subjectOne.getTitle());
        }
        Subject subjectTwo = subjectService.getById(course.getSubjectId());
        if(subjectTwo != null) {
            course.getParam().put("subjectTitle",subjectTwo.getTitle());
        }
        return course;
    }
}
```

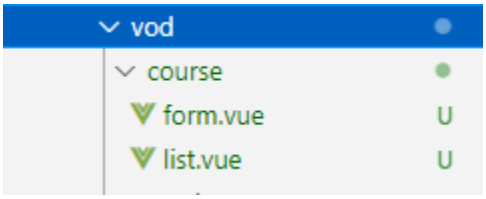
3.2、开发课程列表前端

（1）src 目录下 index.js 文件添加路由

```
// 课程管理
{
  path: '/vod',
  component: Layout,
  redirect: '/vod/course/list',
  name: 'Vod',
  meta: {
    title: '点播管理',
    icon: 'el-icon-bank-card'
  },
  alwaysShow: true,
  children: [
    {
      path: 'course/list',
      name: 'CourseList',
      component: () => import('@/views/vod/course/list'),
      meta: { title: '课程列表' }
    },
    {
      path: 'course/info',
      name: 'CourseInfo',
      component: () => import('@/views/vod/course/form'),
      meta: { title: '发布课程' },
      hidden: true
    },
    {
      path: 'course/info/:id',
      name: 'CourseInfoEdit',
      component: () => import('@/views/vod/course/form'),
      meta: { title: '编辑课程' },
      hidden: true
    },
    {
      path: 'course/chapter/:id',
      name: 'CourseChapterEdit',
      component: () => import('@/views/vod/course/form'),
      meta: { title: '编辑大纲' },
```

```
      hidden: true
    },
    {
      path: 'course/chart/:id',
      name: 'CourseChart',
      component: () => import('@/views/vod/course/chart'),
      meta: { title: '课程统计' },
      hidden: true
    }
  ]
},
```

（2）创建 vue 页面



（3）在 api 目录创建 course.js 文件

```
import request from "@/utils/request";

const api_name = "/admin/vod/course";

export default {
  //课程列表
  getPageList(page, limit, searchObj) {
    return request({
      url: `${api_name}/${page}/${limit}`,
      method: "get",
      params: searchObj,
    });
  },
};
```

（4）在 api 目录 teacher.js 文件定义接口

```
import request from "@/utils/request";

const api_name = "/admin/vod/teacher";

export default {
  //所有讲师
  list() {
    return request({
      url: `${api_name}/findAll`,
      method: `get`,
    });
  },
};
```

（5）编写 list.vue 页面

```
<template>
  <div class="app-container">
    <!--查询表单-->
    <el-card class="operate-container" shadow="never">
      <el-form :inline="true" class="demo-form-inline">
        <!-- 所属分类：级联下拉列表 -->
        <!-- 一级分类 -->
        <el-form-item label="课程类别">
          <el-select
            v-model="searchObj.subjectParentId"
            placeholder="请选择"
            @change="subjectLevelOneChanged"
          >
            <el-option
              v-for="subject in subjectList"
              :key="subject.id"
              :label="subject.title"
              :value="subject.id"
            />
          </el-select>

          <!-- 二级分类 -->
          <el-select v-model="searchObj.subjectId" placeholder="请选择">
            <el-option
              v-for="subject in subjectLevelTwoList"
              :key="subject.id"
              :label="subject.title"
              :value="subject.id"
            />
          </el-select>
        </el-form-item>

        <!-- 标题 -->
        <el-form-item label="标题">
          <el-input v-model="searchObj.title" placeholder="课程标题" />
        </el-form-item>
        <!-- 讲师 -->
```

```
      <el-form-item label="讲师">
        <el-select v-model="searchObj.teacherId" placeholder="请选择讲师">
          <el-option
            v-for="teacher in teacherList"
            :key="teacher.id"
            :label="teacher.name"
            :value="teacher.id"
          />
        </el-select>
      </el-form-item>

      <el-button type="primary" icon="el-icon-search" @click="fetchData()"
        >查询</el-button
      >
      <el-button type="default" @click="resetData()">清空</el-button>
    </el-form>
  </el-card>

  <!-- 工具按钮 -->
  <el-card class="operate-container" shadow="never">
    <i class="el-icon-tickets" style="margin-top: 5px"></i>
    <span style="margin-top: 5px">数据列表</span>
    <el-button class="btn-add" @click="add()">添加</el-button>
  </el-card>

  <!-- 表格 -->
  <el-table :data="list" border stripe>
    <el-table-column label="#" width="50">
      <template slot-scope="scope">
        {{ (page - 1) * limit + scope.$index + 1 }}
      </template>
    </el-table-column>

    <el-table-column label="封面" width="200" align="center">
      <template slot-scope="scope">
        <img :src="scope.row.cover" alt="scope.row.title" width="100%" />
      </template>
```

```
</el-table-column>
<el-table-column label="课程信息">
  <template slot-scope="scope">
    <a href="">{{ scope.row.title }}</a>
    <p>
      分类：{{ scope.row.param.subjectParentTitle }} > {{
      scope.row.param.subjectTitle }}
    </p>
    <p>
      课时：{{ scope.row.lessonNum }} / 浏览：{{ scope.row.viewCount }} /
      付费学员：{{ scope.row.buyCount }}
    </p>
  </template>
</el-table-column>
<el-table-column label="讲师" width="100" align="center">
  <template slot-scope="scope">
    {{ scope.row.param.teacherName }}
  </template>
</el-table-column>
<el-table-column label="价格(元)" width="100" align="center">
  <template slot-scope="scope">
    <!-- {{ typeof '0' }}  {{ typeof 0 }} {{ '0' == 0 }} -->
    <!-- {{ typeof scope.row.price }}
    {{ typeof Number(scope.row.price) }}
    {{ typeof Number(scope.row.price).toFixed(2) }} -->

    <el-tag v-if="Number(scope.row.price) === 0" type="success"
      >免费</el-tag
    >

    <!-- 前端解决保留两位小数的问题 -->
    <!-- <el-tag v-else>{{ Number(scope.row.price).toFixed(2) }}</el-tag> -->

    <!-- 后端解决保留两位小数的问题，前端不用处理 -->
    <el-tag v-else>{{ scope.row.price }}</el-tag>
  </template>
</el-table-column>
```

```
<el-table-column
  prop="status"
  label="课程状态"
  width="100"
  align="center"
>
  <template slot-scope="scope">
    <el-tag :type="scope.row.status === 0 ? 'warning' : 'success'"
      >{{ scope.row.status === 0 ? "未发布" : "已发布" }}</el-tag
    >
  </template>
</el-table-column>
<el-table-column label="发布时间" width="140" align="center">
  <template slot-scope="scope">
    {{ scope.row.createTime ? scope.row.createTime.substr(0, 16) : "" }}
  </template>
</el-table-column>

<el-table-column label="操作" width="210" align="center">
  <template slot-scope="scope">
    <router-link :to="'/vodcourse/course/info/' + scope.row.id">
      <el-button type="text" icon="el-icon-edit">修改</el-button>
    </router-link>
    <router-link :to="'/vodcourse/course/chapter/' + scope.row.id">
      <el-button type="text" icon="el-icon-edit">编辑大纲</el-button>
    </router-link>
    <router-link :to="'/vodcourse/course/chart/' + scope.row.id">
      <el-button type="text" icon="el-icon-edit">课程统计</el-button>
    </router-link>
    <el-button
      type="text"
      icon="el-icon-delete"
      @click="removeById(scope.row.id)"
      >删除</el-button
    >
  </template>
</el-table-column>
```

```
    </el-table>

    <!-- 分页组件 -->
    <el-pagination
      :current-page="page"
      :total="total"
      :page-size="limit"
      :page-sizes="[5, 10, 20, 30, 40, 50, 100]"
      style="padding: 30px 0; text-align: center;"
      layout="total, sizes, prev, pager, next, jumper"
      @size-change="changePageSize"
      @current-change="changeCurrentPage"
    />
  </div>
</template>

<script>
  import courseApi from "@/api/vod/course";
  import teacherApi from "@/api/vod/teacher";
  import subjectApi from "@/api/vod/subject";

  export default {
    data() {
      return {
        list: [], // 课程列表
        total: 0, // 总记录数
        page: 1, // 页码
        limit: 10, // 每页记录数
        searchObj: {
          subjectId: "", // 解决查询表单无法选中二级类别
        }, // 查询条件
        teacherList: [], // 讲师列表
        subjectList: [], // 一级分类列表
        subjectLevelTwoList: [], // 二级分类列表,
      };
    },
```

```
created() {
  this.fetchData();
  // 初始化分类列表
  this.initSubjectList();
  // 获取讲师列表
  this.initTeacherList();
},

methods: {
  fetchData() {
    courseApi
      .getPageList(this.page, this.limit, this.searchObj)
      .then((response) => {
        this.list = response.data.records;
        console.log(this.list);
        this.total = response.data.totalCount;
      });
  },

  initTeacherList() {
    teacherApi.list().then((response) => {
      this.teacherList = response.data;
    });
  },

  initSubjectList() {
    subjectApi.getChildList(0).then((response) => {
      this.subjectList = response.data;
    });
  },

  subjectLevelOneChanged(value) {
    subjectApi.getChildList(value).then((response) => {
      this.subjectLevelTwoList = response.data;
      this.searchObj.subjectId = "";
    });
  },
```

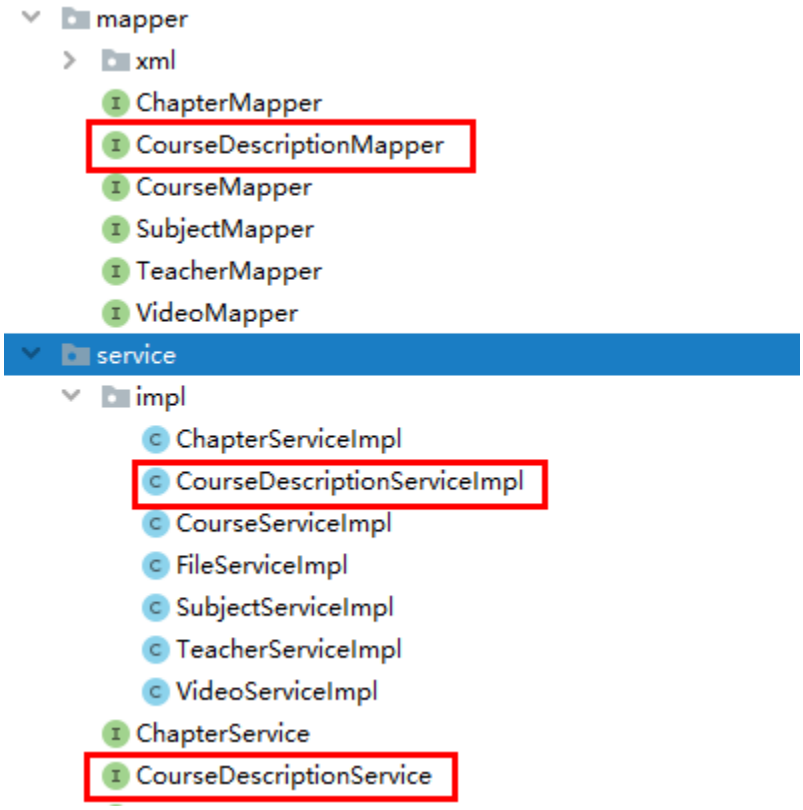
```
      add() {
        this.$router.push({ path: "/vod/course/info" });
      },

      // 每页记录数改变，size：回调参数，表示当前选中的“每页条数”
      changePageSize(size) {
        this.limit = size;
        this.fetchData();
      },

      // 改变页码，page：回调参数，表示当前选中的“页码”
      changeCurrentPage(page) {
        this.page = page;
        this.fetchData();
      },

      // 重置表单
      resetData() {
        this.searchObj = {};
        this.subjectLevelTwoList = []; // 二级分类列表
        this.fetchData();
      },
    },
  };
</script>
```

二、发布课程-填写课程基本信息

1、界面效果

2、添加课程基本信息接口

2.1、创建课程描述的 service 和 mapper



2.2、创建添加课程基本信息接口

（1）controller

```
//添加课程基本信息
@ApiOperation(value = "新增")
@PostMapping("save")
public Result save(@RequestBody CourseFormVo courseFormVo) {
    Long courseId = courseService.saveCourseInfo(courseFormVo);
    return Result.ok(courseId);
}
```

（2）service

```
//实现方法
//添加课程基本信息
@Override
public Long saveCourseInfo(CourseFormVo courseFormVo) {
    //保存课程基本信息
    Course course = new Course();
    BeanUtils.copyProperties(courseFormVo, course);
    baseMapper.insert(course);

    //保存课程详情信息
    CourseDescription courseDescription = new CourseDescription();
    courseDescription.setDescription(courseFormVo.getDescription());
    courseDescription.setCourseId(course.getId());
    descriptionService.save(courseDescription);

    //返回课程id
    return course.getId();
}
```

3、添加课程基本信息前端

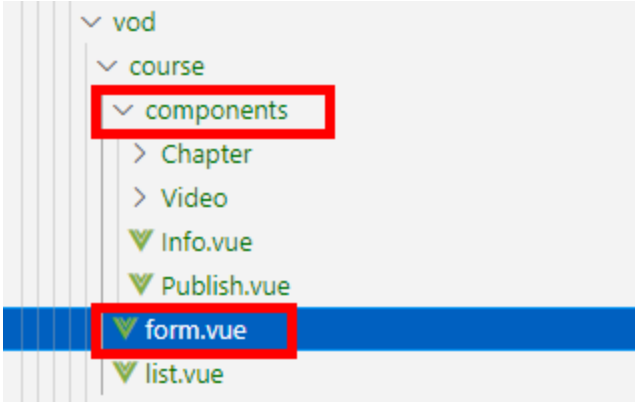
3.1、课程列表 list.vue 添加方法

```
add() {
    this.$router.push({ path: '/vod/course/info' })
},
```

3.2、course.js 定义接口

```
//添加课程基本信息
saveCourseInfo(courseInfo) {
  return request({
    url: `${api_name}/save`,
    method: 'post',
    data: courseInfo
  })
},
```

3.3、创建课程基本信息添加页面



（1）form.vue

```vue
<template>
  <div class="app-container">
    <h2 style="text-align: center;">发布新课程</h2>
    <el-steps
      :active="active"
      finish-status="success"
      simple
      style="margin-bottom: 40px"
    >
      <el-step title="填写课程基本信息" />
      <el-step title="创建课程大纲" />
      <el-step title="发布课程" />
    </el-steps>
    <!-- 填写课程基本信息 -->
    <info v-if="active === 0" />
    <!-- 创建课程大纲 -->
    <chapter v-if="active === 1" />
    <!-- 发布课程 -->
    <Publish v-if="active === 2 || active === 3" />
  </div>
</template>

<script>
  // 引入子组件
  import Info from "@/views/vod/course/components/Info";
  import Chapter from "@/views/vod/course/components/Chapter";
  import Publish from "@/views/vod/course/components/Publish";

  export default {
    components: { Info, Chapter, Publish }, // 注册子组件
    data() {
      return {
        active: 0,
        courseId: null,
      };
    },
    created() {
```

```
      // 获取路由id
      if (this.$route.params.id) {
        this.courseId = this.$route.params.id;
      }
      if (this.$route.name === "CourseInfoEdit") {
        this.active = 0;
      }
      if (this.$route.name === "CourseChapterEdit") {
        this.active = 1;
      }
    },
  };
</script>
```

（2）Info.vue

```
<template>
  <div class="app-container">
    <!-- 课程信息表单 -->
    <el-form label-width="120px">
      <el-form-item label="课程标题">
        <el-input
          v-model="courseInfo.title"
          placeholder=" 示例：机器学习项目课：从基础到搭建项目视频课程。专业名称注意大小写"
        />
      </el-form-item>
      <!-- 课程讲师 -->
      <el-form-item label="课程讲师">
        <el-select v-model="courseInfo.teacherId" placeholder="请选择">
          <el-option
            v-for="teacher in teacherList"
            :key="teacher.id"
            :label="teacher.name"
            :value="teacher.id"
          />
        </el-select>
      </el-form-item>
      <!-- 所属分类：级联下拉列表 -->
      <el-form-item label="课程类别">
        <!-- 一级分类 -->
        <el-select
          v-model="courseInfo.subjectParentId"
          placeholder="请选择"
          @change="subjectChanged"
        >
          <el-option
            v-for="subject in subjectList"
            :key="subject.id"
            :label="subject.title"
            :value="subject.id"
          />
        </el-select>
        <!-- 二级分类 -->
```

```
      <el-select v-model="courseInfo.subjectId" placeholder="请选择">
        <el-option
          v-for="subject in subjectLevelTwoList"
          :key="subject.id"
          :label="subject.title"
          :value="subject.id"
        />
      </el-select>
    </el-form-item>
    <el-form-item label="总课时">
      <el-input-number
        :min="0"
        v-model="courseInfo.lessonNum"
        controls-position="right"
        placeholder="请填写课程的总课时数"
      />
    </el-form-item>
    <!-- 课程简介-->
    <el-form-item label="课程简介">
      <el-input v-model="courseInfo.description" type="textarea" rows="5" />
    </el-form-item>
    <!-- 课程封面 -->
    <el-form-item label="课程封面">
      <el-upload
        :show-file-list="false"
        :on-success="handleCoverSuccess"
        :before-upload="beforeCoverUpload"
        :on-error="handleCoverError"
        :action="BASE_API + '/admin/vod/file/upload?module=cover'"
        class="cover-uploader"
      >
        <img v-if="courseInfo.cover" :src="courseInfo.cover" />
        <i v-else class="el-icon-plus avatar-uploader-icon" />
      </el-upload>
    </el-form-item>
    <el-form-item label="课程价格">
      <el-input-number
```

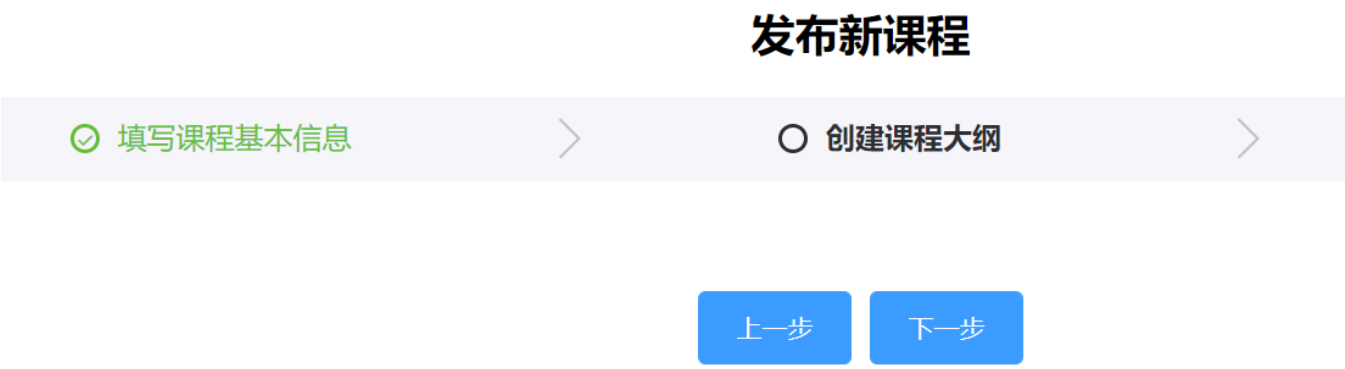
```
          :min="0"
          v-model="courseInfo.price"
          controls-position="right"
          placeholder="免费课程请设置为0元"
        />
        元
      </el-form-item>
    </el-form>

    <div style="text-align:center">
      <el-button
        :disabled="saveBtnDisabled"
        type="primary"
        @click="saveAndNext()"
        >保存并下一步</el-button
      >
    </div>
  </div>
</template>
<script>
  import courseApi from "@/api/vod/course";
  import teacherApi from "@/api/vod/teacher";
  import subjectApi from "@/api/vod/subject";

  export default {
    data() {
      return {
        BASE_API: "http://localhost:8301",
        saveBtnDisabled: false, // 按钮是否禁用
        courseInfo: {
          // 表单数据
          price: 0,
          lessonNum: 0,
          // 以下解决表单数据不全时insert语句非空校验
          teacherId: "",
          subjectId: "",
          subjectParentId: "",
```

```
        cover: "",
        description: "",
      },
      teacherList: [], // 讲师列表
      subjectList: [], // 一级分类列表
      subjectLevelTwoList: [], // 二级分类列表
    };
  },
  created() {
    // 初始化分类列表
    this.initSubjectList();
    // 获取讲师列表
    this.initTeacherList();
  },
  methods: {
    // 获取讲师列表
    initTeacherList() {
      teacherApi.list().then((response) => {
        this.teacherList = response.data;
      });
    },

    // 初始化分类列表
    initSubjectList() {
      subjectApi.getChildList(0).then((response) => {
        this.subjectList = response.data;
      });
    },

    // 选择一级分类，切换二级分类
    subjectChanged(value) {
      subjectApi.getChildList(value).then((response) => {
        this.courseInfo.subjectId = "";
        this.subjectLevelTwoList = response.data;
      });
    },
```

```
// 上传成功回调
handleCoverSuccess(res, file) {
  this.courseInfo.cover = res.data;
},

// 上传校验
beforeCoverUpload(file) {
  const isJPG = file.type === "image/jpeg";
  const isLt2M = file.size / 1024 / 1024 < 2;

  if (!isJPG) {
    this.$message.error("上传头像图片只能是 JPG 格式!");
  }
  if (!isLt2M) {
    this.$message.error("上传头像图片大小不能超过 2MB!");
  }
  return isJPG && isLt2M;
},

// 错误处理
handleCoverError() {
  console.log("error");
  this.$message.error("上传失败2");
},

// 保存并下一步
saveAndNext() {
  this.saveBtnDisabled = true;
  if (!this.$parent.courseId) {
    this.saveData();
  } else {
    //this.updateData()
  }
},

// 保存
saveData() {
```

```
        courseApi.saveCourseInfo(this.courseInfo).then((response) => {
          this.$message.success(response.message);
          this.$parent.courseId = response.data; // 获取courseId
          this.$parent.active = 1; // 下一步
        });
      },
    },
  };
</script>
```

三、发布课程-修改课程基本信息



1、修改课程基本信息接口

1.1、CourseService 定义方法

```
//根据id获取课程信息
CourseFormVo getCourseFormVoById(Long id);

//根据id修改课程信息
void updateCourseById(CourseFormVo courseFormVo);
```

1.2、CourseServiceImpl 实现方法

```
//根据id获取课程信息
@Override
public CourseFormVo getCourseFormVoById(Long id) {
    //从course表中取数据
    Course course = baseMapper.selectById(id);
    if(course == null){
        return null;
    }
    //从course_description表中取数据
    CourseDescription courseDescription = descriptionService.getById(id);
    //创建courseInfoForm对象
    CourseFormVo courseFormVo = new CourseFormVo();
    BeanUtils.copyProperties(course, courseFormVo);
    if(courseDescription != null){
        courseFormVo.setDescription(courseDescription.getDescription());
    }
    return courseFormVo;
}

//根据id修改课程信息
@Override
public void updateCourseById(CourseFormVo courseFormVo) {
    //修改课程基本信息
    Course course = new Course();
    BeanUtils.copyProperties(courseFormVo, course);
    baseMapper.updateById(course);
    //修改课程详情信息
    CourseDescription courseDescription = descriptionService.getById(course.getId());
    courseDescription.setDescription(courseFormVo.getDescription());
    description.setId(course.getId());
    descriptionService.updateById(courseDescription);
}
```

1.3、CourseController 实现方法

```
@ApiOperation(value = "获取")
@GetMapping("get/{id}")
public Result get(@PathVariable Long id) {
    CourseFormVo course = courseService.getCourseFormVoById(id);
    return Result.ok(course);
}

@ApiOperation(value = "修改")
@PutMapping("update")
public Result updateById(@RequestBody CourseFormVo courseFormVo) {
    courseService.updateCourseById(courseFormVo);
    return Result.ok();
}
```

2、修改课程基本信息前端

2.1、course.js 定义方法

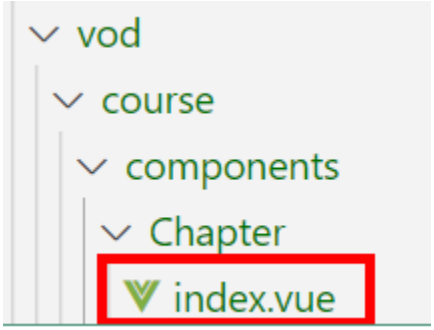
```
//id获取课程信息
getCourseInfoById(id) {
  return request({
    url: `${api_name}/get/${id}`,
    method: 'get'
  })
},
//修改课程信息
updateCourseInfoById(courseInfo) {
  return request({
    url: `${api_name}/update`,
    method: 'put',
    data: courseInfo
  })
},
```

2.2、修改 Info.vue 页面

```
<script>
  created() {
      if (this.$parent.courseId) { // 回显
        this.fetchCourseInfoById(this.$parent.courseId)
      } else { // 新增
        // 初始化分类列表
        this.initSubjectList()
      }
      // 获取讲师列表
      this.initTeacherList()
    },

    methods: {
      // 获取课程信息
      fetchCourseInfoById(id) {
        courseApi.getCourseInfoById(id).then(response => {
          this.courseInfo = response.data
          // 初始化分类列表
          subjectApi.getChildList(0).then(response => {
            this.subjectList = response.data
            // 填充二级菜单：遍历一级菜单列表，
            this.subjectList.forEach(subject => {
              // 找到和courseInfo.subjectParentId一致的父类别记录
              if (subject.id === this.courseInfo.subjectParentId) {
                // 拿到当前类别下的子类别列表，将子类别列表填入二级下拉菜单列表
                subjectApi.getChildList(subject.id).then(response => {
                  this.subjectLevelTwoList = response.data
                })
              }
            })
          })
        })
      },
      // 保存并下一步
      saveAndNext() {
        this.saveBtnDisabled = true
        if (!this.$parent.courseId) {
```

```
          this.saveData()
        } else {
          this.updateData()
        }
      },
      // 修改
      updateData() {
        courseApi.updateCourseInfoById(this.courseInfo).then(response => {
          this.$message.success(response.message)
          this.$parent.courseId = response.data // 获取courseId
          this.$parent.active = 1 // 下一步
        })
      },
    }
</script>
```

2.3、创建 Chapter-index.vue 页面



```vue
<template>
  <div class="app-container">
    <div style="text-align:center">
      <el-button type="primary" @click="prev()">上一步</el-button>
      <el-button type="primary" @click="next()">下一步</el-button>
    </div>
  </div>
</template>
<script>
  export default {
    data() {
      return {};
    },
    created() {},
    methods: {
      // 上一步
      prev() {
        this.$parent.active = 0;
      },
      // 下一步
      next() {
        this.$parent.active = 2;
      },
    },
  };
</script>
```