src/main/java/com/iot/courseManager/controller/CourseController.java

```
package com.ruoyi.courseManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.courseManager.domain.Course;
import com.ruoyi.courseManager.service.ICourseService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
 * 课程管理 Controller
 *
```

```
 * @author LJH
 * @date 2024-11-26
 */
@RestController
@RequestMapping("/courseManager/course")
public class CourseController extends BaseController
{
    @Autowired
    private ICourseService courseService;

    /**
     * 查询课程管理列表
     */
    @PreAuthorize("@ss.hasPermi('courseManager:course:list')")
    @GetMapping("/list")
    public TableDataInfo list(Course course)
    {
        startPage();
        List<Course> list = courseService.selectCourseList(course);
        return getDataTable(list);
    }

    /**
     * 导出课程管理列表
     */
    @PreAuthorize("@ss.hasPermi('courseManager:course:export')")
    @Log(title = "课程管理", businessType = BusinessType.EXPORT)
```

```
@PostMapping("/export")
public void export(HttpServletResponse response, Course course)
{
    List<Course> list = courseService.selectCourseList(course);
    ExcelUtil<Course> util = new ExcelUtil<Course>(Course.class);
    util.exportExcel(response, list, "课程管理数据");
}

/**
 * 获取课程管理详细信息
 */
@PreAuthorize("@ss.hasPermi('courseManager:course:query')")
@GetMapping(value = "/{courseId}")
public AjaxResult getInfo(@PathVariable("courseId") Long courseId)
{
    return success(courseService.selectCourseByCourseId(courseId));
}

/**
 * 新增课程管理
 */
@PreAuthorize("@ss.hasPermi('courseManager:course:add')")
@Log(title = "课程管理", businessType = BusinessType.INSERT)
@PostMapping
public AjaxResult add(@RequestBody Course course)
{
    return toAjax(courseService.insertCourse(course));
```

```
    }

    /**
     * 修改课程管理
     */
    @PreAuthorize("@ss.hasPermi('courseManager:course:edit')")
    @Log(title = "课程管理", businessType = BusinessType.UPDATE)
    @PutMapping
    public AjaxResult edit(@RequestBody Course course)
    {
        return toAjax(courseService.updateCourse(course));
    }

    /**
     * 删除课程管理
     */
    @PreAuthorize("@ss.hasPermi('courseManager:course:remove')")
    @Log(title = "课程管理", businessType = BusinessType.DELETE)
    @DeleteMapping("/{courseIds}")
    public AjaxResult remove(@PathVariable Long[] courseIds)
    {
        return toAjax(courseService.deleteCourseByCourseIds(courseIds));
    }
}
```

src/main/java/com/iot/courseManager/domain/Course.java

```
package com.ruoyi.courseManager.domain;

import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 课程管理对象  course
 *
 * @author LJH
 * @date 2024-11-26
 */
public class Course extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 课程 id */
    private Long courseId;

    /** 教师 id */
    @Excel(name = "教师 id")
    private Long teaId;

    /** 课程号 */
    @Excel(name = "课程号")
    private String courseNum;
```

```
/** 课程名 */
@Excel(name = "课程名")
private String courseName;

public void setCourseId(Long courseId)
{
    this.courseId = courseId;
}

public Long getCourseId()
{
    return courseId;
}
public void setTeaId(Long teaId)
{
    this.teaId = teaId;
}

public Long getTeaId()
{
    return teaId;
}
public void setCourseNum(String courseNum)
{
    this.courseNum = courseNum;
}
```

```
    public String getCourseNum()
    {
        return courseNum;
    }
    public void setCourseName(String courseName)
    {
        this.courseName = courseName;
    }

    public String getCourseName()
    {
        return courseName;
    }

    @Override
    public String toString() {
        return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
                .append("courseId", getCourseId())
                .append("teaId", getTeaId())
                .append("courseNum", getCourseNum())
                .append("courseName", getCourseName())
                .toString();
    }
}
```

src/main/java/com/iot/courseManager/mapper/CourseMapper.java

```
package com.ruoyi.courseManager.mapper;

import java.util.List;
import com.ruoyi.courseManager.domain.Course;

/**
 * 课程管理 Mapper 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface CourseMapper
{
    /**
     * 查询课程管理
     *
     * @param courseId 课程管理主键
     * @return 课程管理
     */
    public Course selectCourseByCourseId(Long courseId);

    /**
     * 查询课程管理列表
     *
     * @param course 课程管理
     * @return 课程管理集合
```

```
 */
public List<Course> selectCourseList(Course course);

/**
 * 新增课程管理
 *
 * @param course 课程管理
 * @return 结果
 */
public int insertCourse(Course course);

/**
 * 修改课程管理
 *
 * @param course 课程管理
 * @return 结果
 */
public int updateCourse(Course course);

/**
 * 删除课程管理
 *
 * @param courseId 课程管理主键
 * @return 结果
 */
public int deleteCourseByCourseId(Long courseId);
```

```
    /**
     * 批量删除课程管理
     *
     * @param courseIds 需要删除的数据主键集合
     * @return 结果
     */
    public int deleteCourseByCourseIds(Long[] courseIds);
}
```

src/main/java/com/iot/courseManager/service/ICourseService.java

```
package com.ruoyi.courseManager.service;

import java.util.List;
import com.ruoyi.courseManager.domain.Course;

/**
 * 课程管理 Service 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface ICourseService
{
    /**
     * 查询课程管理
     *
```

```
     * @param courseId 课程管理主键
     * @return 课程管理
     */
    public Course selectCourseByCourseId(Long courseId);

    /**
     * 查询课程管理列表
     *
     * @param course 课程管理
     * @return 课程管理集合
     */
    public List<Course> selectCourseList(Course course);

    /**
     * 新增课程管理
     *
     * @param course 课程管理
     * @return 结果
     */
    public int insertCourse(Course course);

    /**
     * 修改课程管理
     *
     * @param course 课程管理
     * @return 结果
     */
```

```
    public int updateCourse(Course course);

    /**
     * 批量删除课程管理
     *
     * @param courseIds 需要删除的课程管理主键集合
     * @return 结果
     */
    public int deleteCourseByCourseIds(Long[] courseIds);

    /**
     * 删除课程管理信息
     *
     * @param courseId 课程管理主键
     * @return 结果
     */
    public int deleteCourseByCourseId(Long courseId);
}
```

src/main/java/com/iot/courseManager/service/impl/CourseServiceImpl.java

```
package com.ruoyi.courseManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.courseManager.mapper.CourseMapper;
```

```
import com.ruoyi.courseManager.domain.Course;
import com.ruoyi.courseManager.service.ICourseService;

/**
 * 课程管理 Service 业务层处理
 *
 * @author LJH
 * @date 2024-11-26
 */
@Service
public class CourseServiceImpl implements ICourseService
{
    @Autowired
    private CourseMapper courseMapper;

    /**
     * 查询课程管理
     *
     * @param courseId 课程管理主键
     * @return 课程管理
     */
    @Override
    public Course selectCourseByCourseId(Long courseId)
    {
        return courseMapper.selectCourseByCourseId(courseId);
    }
```

```
/**
 * 查询课程管理列表
 *
 * @param course 课程管理
 * @return 课程管理
 */
@Override
public List<Course> selectCourseList(Course course)
{
    return courseMapper.selectCourseList(course);
}

/**
 * 新增课程管理
 *
 * @param course 课程管理
 * @return 结果
 */
@Override
public int insertCourse(Course course)
{
    return courseMapper.insertCourse(course);
}

/**
 * 修改课程管理
 *
```

```
     * @param course  课程管理
     * @return  结果
     */
    @Override
    public int updateCourse(Course course)
    {
        return courseMapper.updateCourse(course);
    }

    /**
     *  批量删除课程管理
     *
     * @param courseIds  需要删除的课程管理主键
     * @return  结果
     */
    @Override
    public int deleteCourseByCourseIds(Long[] courseIds)
    {
        return courseMapper.deleteCourseByCourseIds(courseIds);
    }

    /**
     *  删除课程管理信息
     *
     * @param courseId  课程管理主键
     * @return  结果
     */
```

```
    @Override
    public int deleteCourseByCourseId(Long courseId)
    {
        return courseMapper.deleteCourseByCourseId(courseId);
    }
}
```

src/main/java/com/iot/machineManager/controller/MachineController.java

```
package com.ruoyi.machineManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.machineManager.domain.Machine;
```

```
import com.ruoyi.machineManager.service.IMachineService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
 * 机器管理 Controller
 *
 * @author LJH
 * @date 2024-11-26
 */
@RestController
@RequestMapping("/machineManager/machine")
public class MachineController extends BaseController
{
    @Autowired
    private IMachineService machineService;

    /**
     * 查询机器管理列表
     */
    @PreAuthorize("@ss.hasPermi('machineManager:machine:list')")
    @GetMapping("/list")
    public TableDataInfo list(Machine machine)
    {
        startPage();
        List<Machine> list = machineService.selectMachineList(machine);
        return getDataTable(list);
```

```
}

/**
 * 导出机器管理列表
 */
@PreAuthorize("@ss.hasPermi('machineManager:machine:export')")
@Log(title = "机器管理", businessType = BusinessType.EXPORT)
@PostMapping("/export")
public void export(HttpServletResponse response, Machine machine)
{
    List<Machine> list = machineService.selectMachineList(machine);
    ExcelUtil<Machine> util = new ExcelUtil<Machine>(Machine.class);
    util.exportExcel(response, list, "机器管理数据");
}

/**
 * 获取机器管理详细信息
 */
@PreAuthorize("@ss.hasPermi('machineManager:machine:query')")
@GetMapping(value = "/{macId}")
public AjaxResult getInfo(@PathVariable("macId") Long macId)
{
    return success(machineService.selectMachineByMacId(macId));
}

/**
 * 新增机器管理
```

```
 */
@PreAuthorize("@ss.hasPermi('machineManager:machine:add')")
@Log(title = "机器管理", businessType = BusinessType.INSERT)
@PostMapping
public AjaxResult add(@RequestBody Machine machine)
{
    return toAjax(machineService.insertMachine(machine));
}

/**
 * 修改机器管理
 */
@PreAuthorize("@ss.hasPermi('machineManager:machine:edit')")
@Log(title = "机器管理", businessType = BusinessType.UPDATE)
@PutMapping
public AjaxResult edit(@RequestBody Machine machine)
{
    return toAjax(machineService.updateMachine(machine));
}

/**
 * 删除机器管理
 */
@PreAuthorize("@ss.hasPermi('machineManager:machine:remove')")
@Log(title = "机器管理", businessType = BusinessType.DELETE)
@DeleteMapping("/{macIds}")
public AjaxResult remove(@PathVariable Long[] macIds)
```

```
    {
        return toAjax(machineService.deleteMachineByMacIds(macIds));
    }
}
```

src/main/java/com/iot/machineManager/domain/Machine.java

```
package com.ruoyi.machineManager.domain;

import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 机器管理对象 machine
 *
 * @author LJH
 * @date 2024-11-26
 */
public class Machine extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 机器 id */
    private Long macId;
```

```
/** 机器编号 */
@Excel(name = "机器编号")
private String macNum;

/** 机器名称 */
@Excel(name = "机器名称")
private String macName;

/** 机器类型 */
@Excel(name = "机器类型")
private String macType;

public void setMacId(Long macId)
{
    this.macId = macId;
}

public Long getMacId()
{
    return macId;
}
public void setMacNum(String macNum)
{
    this.macNum = macNum;
}

public String getMacNum()
```

```
{
        return macNum;
}
public void setMacName(String macName)
{
        this.macName = macName;
}

public String getMacName()
{
        return macName;
}
public void setMacType(String macType)
{
        this.macType = macType;
}

public String getMacType()
{
        return macType;
}

@Override
public String toString() {
        return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
                .append("macId", getMacId())
                .append("macNum", getMacNum())
```

```
                .append("macName", getMacName())
                .append("macType", getMacType())
                .toString();
    }
}
```

src/main/java/com/iot/machineManager/mapper/MachineMapper.java

```
package com.ruoyi.machineManager.mapper;

import java.util.List;
import com.ruoyi.machineManager.domain.Machine;

/**
 * 机器管理 Mapper 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface MachineMapper
{
    /**
     * 查询机器管理
     *
     * @param macId 机器管理主键
     * @return 机器管理
     */
```

```
public Machine selectMachineByMacId(Long macId);

/**
 * 查询机器管理列表
 *
 * @param machine 机器管理
 * @return 机器管理集合
 */
public List<Machine> selectMachineList(Machine machine);

/**
 * 新增机器管理
 *
 * @param machine 机器管理
 * @return 结果
 */
public int insertMachine(Machine machine);

/**
 * 修改机器管理
 *
 * @param machine 机器管理
 * @return 结果
 */
public int updateMachine(Machine machine);

/**
```

```
     * 删除机器管理
     *
     * @param macId 机器管理主键
     * @return 结果
     */
    public int deleteMachineByMacId(Long macId);

    /**
     * 批量删除机器管理
     *
     * @param macIds 需要删除的数据主键集合
     * @return 结果
     */
    public int deleteMachineByMacIds(Long[] macIds);
}
```

src/main/java/com/iot/machineManager/service/IMachineService.java

```
package com.ruoyi.machineManager.service;

import java.util.List;
import com.ruoyi.machineManager.domain.Machine;

/**
 * 机器管理 Service 接口
 *
 * @author LJH
```

```
 * @date 2024-11-26
 */
public interface IMachineService
{
    /**
     * 查询机器管理
     *
     * @param macId 机器管理主键
     * @return 机器管理
     */
    public Machine selectMachineByMacId(Long macId);

    /**
     * 查询机器管理列表
     *
     * @param machine 机器管理
     * @return 机器管理集合
     */
    public List<Machine> selectMachineList(Machine machine);

    /**
     * 新增机器管理
     *
     * @param machine 机器管理
     * @return 结果
     */
    public int insertMachine(Machine machine);
```

```
    /**
     * 修改机器管理
     *
     * @param machine 机器管理
     * @return 结果
     */
    public int updateMachine(Machine machine);

    /**
     * 批量删除机器管理
     *
     * @param macIds 需要删除的机器管理主键集合
     * @return 结果
     */
    public int deleteMachineByMacIds(Long[] macIds);

    /**
     * 删除机器管理信息
     *
     * @param macId 机器管理主键
     * @return 结果
     */
    public int deleteMachineByMacId(Long macId);
}
```

src/main/java/com/iot/machineManager/service/impl/MachineServiceImpl.java

```
package com.ruoyi.machineManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.machineManager.mapper.MachineMapper;
import com.ruoyi.machineManager.domain.Machine;
import com.ruoyi.machineManager.service.IMachineService;

/**
 * 机器管理 Service 业务层处理
 *
 * @author LJH
 * @date 2024-11-26
 */
@Service
public class MachineServiceImpl implements IMachineService
{
    @Autowired
    private MachineMapper machineMapper;

    /**
     * 查询机器管理
     *
     * @param macId 机器管理主键
     * @return 机器管理
```

```
 */
@Override
public Machine selectMachineByMacId(Long macId)
{
    return machineMapper.selectMachineByMacId(macId);
}

/**
 * 查询机器管理列表
 *
 * @param machine 机器管理
 * @return 机器管理
 */
@Override
public List<Machine> selectMachineList(Machine machine)
{
    return machineMapper.selectMachineList(machine);
}

/**
 * 新增机器管理
 *
 * @param machine 机器管理
 * @return 结果
 */
@Override
public int insertMachine(Machine machine)
```

```
{
        return machineMapper.insertMachine(machine);
}

/**
 * 修改机器管理
 *
 * @param machine 机器管理
 * @return 结果
 */
@Override
public int updateMachine(Machine machine)
{
        return machineMapper.updateMachine(machine);
}

/**
 * 批量删除机器管理
 *
 * @param macIds 需要删除的机器管理主键
 * @return 结果
 */
@Override
public int deleteMachineByMacIds(Long[] macIds)
{
        return machineMapper.deleteMachineByMacIds(macIds);
}
```

```
    /**
     * 删除机器管理信息
     *
     * @param macId 机器管理主键
     * @return 结果
     */
    @Override
    public int deleteMachineByMacId(Long macId)
    {
        return machineMapper.deleteMachineByMacId(macId);
    }
}
```

src/main/java/com/iot/phytestManager/controller/PhytestController.java

```
package com.ruoyi.phytestManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.phytestManager.domain.Phytest;
import com.ruoyi.phytestManager.service.IPhytestService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
 * 体测项目管理 Controller
 *
 * @author LJH
 * @date 2024-11-26
 */
@RestController
@RequestMapping("/phytestManager/phytest")
public class PhytestController extends BaseController
{
    @Autowired
    private IPhytestService phytestService;

    /**
     * 查询体测项目管理列表
     */
```

```
@PreAuthorize("@ss.hasPermi('phytestManager:phytest:list')")
@GetMapping("/list")
public TableDataInfo list(Phytest phytest)
{
    startPage();
    List<Phytest> list = phytestService.selectPhytestList(phytest);
    return getDataTable(list);
}

/**
 * 导出体测项目管理列表
 */
@PreAuthorize("@ss.hasPermi('phytestManager:phytest:export')")
@Log(title = "体测项目管理", businessType = BusinessType.EXPORT)
@PostMapping("/export")
public void export(HttpServletResponse response, Phytest phytest)
{
    List<Phytest> list = phytestService.selectPhytestList(phytest);
    ExcelUtil<Phytest> util = new ExcelUtil<Phytest>(Phytest.class);
    util.exportExcel(response, list, "体测项目管理数据");
}

/**
 * 获取体测项目管理详细信息
 */
@PreAuthorize("@ss.hasPermi('phytestManager:phytest:query')")
@GetMapping(value = "/{ptId}")
```

```
public AjaxResult getInfo(@PathVariable("ptId") Long ptId)
{
    return success(phytestService.selectPhytestByPtId(ptId));
}

/**
 * 新增体测项目管理
 */
@PreAuthorize("@ss.hasPermi('phytestManager:phytest:add')")
@Log(title = "体测项目管理", businessType = BusinessType.INSERT)
@PostMapping
public AjaxResult add(@RequestBody Phytest phytest)
{
    return toAjax(phytestService.insertPhytest(phytest));
}

/**
 * 修改体测项目管理
 */
@PreAuthorize("@ss.hasPermi('phytestManager:phytest:edit')")
@Log(title = "体测项目管理", businessType = BusinessType.UPDATE)
@PutMapping
public AjaxResult edit(@RequestBody Phytest phytest)
{
    return toAjax(phytestService.updatePhytest(phytest));
}
```

```
    /**
     * 删除体测项目管理
     */
    @PreAuthorize("@ss.hasPermi('phytestManager:phytest:remove')")
    @Log(title = "体测项目管理", businessType = BusinessType.DELETE)
    @DeleteMapping("/{ptIds}")
    public AjaxResult remove(@PathVariable Long[] ptIds)
    {
        return toAjax(phytestService.deletePhytestByPtIds(ptIds));
    }
}
```

src/main/java/com/iot/phytestManager/domain/Phytest.java

```
package com.ruoyi.phytestManager.domain;

import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 体测项目管理对象 phytest
 *
 * @author LJH
 * @date 2024-11-26
 */
```

```
public class Phytest extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 项目 id */
    private Long ptId;

    /** 项目序号 */
    @Excel(name = "项目序号")
    private String ptNum;

    /** 项目名称 */
    @Excel(name = "项目名称")
    private String ptName;

    /** 项目权重 */
    @Excel(name = "项目权重")
    private Long ptWeight;

    public void setPtId(Long ptId)
    {
        this.ptId = ptId;
    }

    public Long getPtId()
    {
        return ptId;
```

```
}
public void setPtNum(String ptNum)
{
	this.ptNum = ptNum;
}

public String getPtNum()
{
	return ptNum;
}
public void setPtName(String ptName)
{
	this.ptName = ptName;
}

public String getPtName()
{
	return ptName;
}
public void setPtWeight(Long ptWeight)
{
	this.ptWeight = ptWeight;
}

public Long getPtWeight()
{
	return ptWeight;
```

```
    }

    @Override
    public String toString() {
        return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
                .append("ptId", getPtId())
                .append("ptNum", getPtNum())
                .append("ptName", getPtName())
                .append("ptWeight", getPtWeight())
                .toString();
    }
}
```

src/main/java/com/iot/phytestManager/mapper/PhytestMapper.java

```
package com.ruoyi.phytestManager.mapper;

import java.util.List;
import com.ruoyi.phytestManager.domain.Phytest;

/**
 * 体测项目管理 Mapper 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface PhytestMapper
```

```
{
    /**
     * 查询体测项目管理
     *
     * @param ptId 体测项目管理主键
     * @return 体测项目管理
     */
    public Phytest selectPhytestByPtId(Long ptId);

    /**
     * 查询体测项目管理列表
     *
     * @param phytest 体测项目管理
     * @return 体测项目管理集合
     */
    public List<Phytest> selectPhytestList(Phytest phytest);

    /**
     * 新增体测项目管理
     *
     * @param phytest 体测项目管理
     * @return 结果
     */
    public int insertPhytest(Phytest phytest);

    /**
     * 修改体测项目管理
```

```
     *
     * @param phytest 体测项目管理
     * @return 结果
     */
    public int updatePhytest(Phytest phytest);

    /**
     * 删除体测项目管理
     *
     * @param ptId 体测项目管理主键
     * @return 结果
     */
    public int deletePhytestByPtId(Long ptId);

    /**
     * 批量删除体测项目管理
     *
     * @param ptIds 需要删除的数据主键集合
     * @return 结果
     */
    public int deletePhytestByPtIds(Long[] ptIds);
}
```

src/main/java/com/iot/phytestManager/service/IPhytestService.java

```
package com.ruoyi.phytestManager.service;
```

```
import java.util.List;
import com.ruoyi.phytestManager.domain.Phytest;

/**
 * 体测项目管理 Service 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface IPhytestService
{
    /**
     * 查询体测项目管理
     *
     * @param ptId 体测项目管理主键
     * @return 体测项目管理
     */
    public Phytest selectPhytestByPtId(Long ptId);

    /**
     * 查询体测项目管理列表
     *
     * @param phytest 体测项目管理
     * @return 体测项目管理集合
     */
    public List<Phytest> selectPhytestList(Phytest phytest);
```

```
/**
 * 新增体测项目管理
 *
 * @param phytest 体测项目管理
 * @return 结果
 */
public int insertPhytest(Phytest phytest);

/**
 * 修改体测项目管理
 *
 * @param phytest 体测项目管理
 * @return 结果
 */
public int updatePhytest(Phytest phytest);

/**
 * 批量删除体测项目管理
 *
 * @param ptIds 需要删除的体测项目管理主键集合
 * @return 结果
 */
public int deletePhytestByPtIds(Long[] ptIds);

/**
 * 删除体测项目管理信息
 *
```

```
     * @param ptId 体测项目管理主键
     * @return 结果
     */
    public int deletePhytestByPtId(Long ptId);
}
```

src/main/java/com/iot/phytestManager/service/impl/PhytestServiceImpl.java

```
package com.ruoyi.phytestManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.phytestManager.mapper.PhytestMapper;
import com.ruoyi.phytestManager.domain.Phytest;
import com.ruoyi.phytestManager.service.IPhytestService;

/**
 * 体测项目管理 Service 业务层处理
 *
 * @author LJH
 * @date 2024-11-26
 */
@Service
public class PhytestServiceImpl implements IPhytestService
{
    @Autowired
```

```
private PhytestMapper phytestMapper;

/**
 * 查询体测项目管理
 *
 * @param ptId 体测项目管理主键
 * @return 体测项目管理
 */
@Override
public Phytest selectPhytestByPtId(Long ptId)
{
    return phytestMapper.selectPhytestByPtId(ptId);
}

/**
 * 查询体测项目管理列表
 *
 * @param phytest 体测项目管理
 * @return 体测项目管理
 */
@Override
public List<Phytest> selectPhytestList(Phytest phytest)
{
    return phytestMapper.selectPhytestList(phytest);
}

/**
```

```
     * 新增体测项目管理
     *
     * @param phytest 体测项目管理
     * @return 结果
     */
    @Override
    public int insertPhytest(Phytest phytest)
    {
        return phytestMapper.insertPhytest(phytest);
    }

    /**
     * 修改体测项目管理
     *
     * @param phytest 体测项目管理
     * @return 结果
     */
    @Override
    public int updatePhytest(Phytest phytest)
    {
        return phytestMapper.updatePhytest(phytest);
    }

    /**
     * 批量删除体测项目管理
     *
     * @param ptIds 需要删除的体测项目管理主键
```

```
     * @return  结果
     */
    @Override
    public int deletePhytestByPtIds(Long[] ptIds)
    {
        return phytestMapper.deletePhytestByPtIds(ptIds);
    }

    /**
     * 删除体测项目管理信息
     *
     * @param ptId  体测项目管理主键
     * @return  结果
     */
    @Override
    public int deletePhytestByPtId(Long ptId)
    {
        return phytestMapper.deletePhytestByPtId(ptId);
    }
}
```

src/main/java/com/iot/ptbookManager/controller/PtbookController.java

```
package com.ruoyi.ptbookManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.ptbookManager.domain.Ptbook;
import com.ruoyi.ptbookManager.service.IPtbookService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
 * 体测预约管理 Controller
 *
 * @author LJH
 * @date 2024-11-26
 */
@RestController
@RequestMapping("/ptbookManager/ptbook")
public class PtbookController extends BaseController
```

```
{
    @Autowired
    private IPtbookService ptbookService;

    /**
     * 查询体测预约管理列表
     */
    @PreAuthorize("@ss.hasPermi('ptbookManager:ptbook:list')")
    @GetMapping("/list")
    public TableDataInfo list(Ptbook ptbook)
    {
        startPage();
        List<Ptbook> list = ptbookService.selectPtbookList(ptbook);
        return getDataTable(list);
    }

    /**
     * 导出体测预约管理列表
     */
    @PreAuthorize("@ss.hasPermi('ptbookManager:ptbook:export')")
    @Log(title = "体测预约管理", businessType = BusinessType.EXPORT)
    @PostMapping("/export")
    public void export(HttpServletResponse response, Ptbook ptbook)
    {
        List<Ptbook> list = ptbookService.selectPtbookList(ptbook);
        ExcelUtil<Ptbook> util = new ExcelUtil<Ptbook>(Ptbook.class);
        util.exportExcel(response, list, "体测预约管理数据");
```

```
}

/**
 * 获取体测预约管理详细信息
 */
@PreAuthorize("@ss.hasPermi('ptbookManager:ptbook:query')")
@GetMapping(value = "/{ptbookId}")
public AjaxResult getInfo(@PathVariable("ptbookId") Long ptbookId)
{
    return success(ptbookService.selectPtbookByPtbookId(ptbookId));
}

/**
 * 新增体测预约管理
 */
@PreAuthorize("@ss.hasPermi('ptbookManager:ptbook:add')")
@Log(title = "体测预约管理", businessType = BusinessType.INSERT)
@PostMapping
public AjaxResult add(@RequestBody Ptbook ptbook)
{
    return toAjax(ptbookService.insertPtbook(ptbook));
}

/**
 * 修改体测预约管理
 */
@PreAuthorize("@ss.hasPermi('ptbookManager:ptbook:edit')")
```

```
    @Log(title = "体测预约管理", businessType = BusinessType.UPDATE)
    @PutMapping
    public AjaxResult edit(@RequestBody Ptbook ptbook)
    {
        return toAjax(ptbookService.updatePtbook(ptbook));
    }

    /**
     * 删除体测预约管理
     */
    @PreAuthorize("@ss.hasPermi('ptbookManager:ptbook:remove')")
    @Log(title = "体测预约管理", businessType = BusinessType.DELETE)
    @DeleteMapping("/{ptbookIds}")
    public AjaxResult remove(@PathVariable Long[] ptbookIds)
    {
        return toAjax(ptbookService.deletePtbookByPtbookIds(ptbookIds));
    }
}
```

src/main/java/com/iot/ptbookManager/domain/Ptbook.java

```
package com.ruoyi.ptbookManager.domain;

import java.util.Date;
import com.fasterxml.jackson.annotation.JsonFormat;
import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
```

```
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 体测预约管理对象 ptbook
 *
 * @author LJH
 * @date 2024-11-26
 */
public class Ptbook extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 预约 id */
    private Long ptbookId;

    /** 容量 */
    @Excel(name = "容量")
    private Long ptbookCon;

    /** 已约人数 */
    @Excel(name = "已约人数")
    private Long ptbookApnum;

    /** 预约开始日期 */
    @JsonFormat(pattern = "yyyy-MM-dd")
    @Excel(name = "预约开始日期", width = 30, dateFormat = "yyyy-MM-dd")
```

```
private Date ptbookBegintime;

/** 测试时间 */
@JsonFormat(pattern = "yyyy-MM-dd")
@Excel(name = "测试时间", width = 30, dateFormat = "yyyy-MM-dd")
private Date ptbookTesttime;

/** 预约结束日期 */
@JsonFormat(pattern = "yyyy-MM-dd")
@Excel(name = "预约结束日期", width = 30, dateFormat = "yyyy-MM-dd")
private Date ptbookEndtime;

public void setPtbookId(Long ptbookId)
{
    this.ptbookId = ptbookId;
}

public Long getPtbookId()
{
    return ptbookId;
}
public void setPtbookCon(Long ptbookCon)
{
    this.ptbookCon = ptbookCon;
}

public Long getPtbookCon()
```

```
{
        return ptbookCon;
}
public void setPtbookApnum(Long ptbookApnum)
{
        this.ptbookApnum = ptbookApnum;
}

public Long getPtbookApnum()
{
        return ptbookApnum;
}
public void setPtbookBegintime(Date ptbookBegintime)
{
        this.ptbookBegintime = ptbookBegintime;
}

public Date getPtbookBegintime()
{
        return ptbookBegintime;
}
public void setPtbookTesttime(Date ptbookTesttime)
{
        this.ptbookTesttime = ptbookTesttime;
}

public Date getPtbookTesttime()
```

```
    {
        return ptbookTesttime;
    }
    public void setPtbookEndtime(Date ptbookEndtime)
    {
        this.ptbookEndtime = ptbookEndtime;
    }

    public Date getPtbookEndtime()
    {
        return ptbookEndtime;
    }

    @Override
    public String toString() {
        return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
                .append("ptbookId", getPtbookId())
                .append("ptbookCon", getPtbookCon())
                .append("ptbookApnum", getPtbookApnum())
                .append("ptbookBegintime", getPtbookBegintime())
                .append("ptbookTesttime", getPtbookTesttime())
                .append("ptbookEndtime", getPtbookEndtime())
                .toString();
    }
}
```

src/main/java/com/iot/ptbookManager/mapper/PtbookMapper.java

```
package com.ruoyi.ptbookManager.mapper;

import java.util.List;
import com.ruoyi.ptbookManager.domain.Ptbook;

/**
 * 体测预约管理 Mapper 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface PtbookMapper
{
    /**
     * 查询体测预约管理
     *
     * @param ptbookId 体测预约管理主键
     * @return 体测预约管理
     */
    public Ptbook selectPtbookByPtbookId(Long ptbookId);

    /**
     * 查询体测预约管理列表
     *
     * @param ptbook 体测预约管理
     * @return 体测预约管理集合
```

```
 */
public List<Ptbook> selectPtbookList(Ptbook ptbook);

/**
 * 新增体测预约管理
 *
 * @param ptbook 体测预约管理
 * @return 结果
 */
public int insertPtbook(Ptbook ptbook);

/**
 * 修改体测预约管理
 *
 * @param ptbook 体测预约管理
 * @return 结果
 */
public int updatePtbook(Ptbook ptbook);

/**
 * 删除体测预约管理
 *
 * @param ptbookId 体测预约管理主键
 * @return 结果
 */
public int deletePtbookByPtbookId(Long ptbookId);
```

```
    /**
     * 批量删除体测预约管理
     *
     * @param ptbookIds 需要删除的数据主键集合
     * @return 结果
     */
    public int deletePtbookByPtbookIds(Long[] ptbookIds);
}
```

src/main/java/com/iot/ptbookManager/service/IPtbookService.java

```
package com.ruoyi.ptbookManager.service;

import java.util.List;
import com.ruoyi.ptbookManager.domain.Ptbook;

/**
 * 体测预约管理 Service 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface IPtbookService
{
    /**
     * 查询体测预约管理
     *
```

```
     * @param ptbookId 体测预约管理主键
     * @return 体测预约管理
     */
    public Ptbook selectPtbookByPtbookId(Long ptbookId);

    /**
     * 查询体测预约管理列表
     *
     * @param ptbook 体测预约管理
     * @return 体测预约管理集合
     */
    public List<Ptbook> selectPtbookList(Ptbook ptbook);

    /**
     * 新增体测预约管理
     *
     * @param ptbook 体测预约管理
     * @return 结果
     */
    public int insertPtbook(Ptbook ptbook);

    /**
     * 修改体测预约管理
     *
     * @param ptbook 体测预约管理
     * @return 结果
     */
```

```
    public int updatePtbook(Ptbook ptbook);

    /**
     * 批量删除体测预约管理
     *
     * @param ptbookIds 需要删除的体测预约管理主键集合
     * @return 结果
     */
    public int deletePtbookByPtbookIds(Long[] ptbookIds);

    /**
     * 删除体测预约管理信息
     *
     * @param ptbookId 体测预约管理主键
     * @return 结果
     */
    public int deletePtbookByPtbookId(Long ptbookId);
}
```

src/main/java/com/iot/ptbookManager/service/impl/PtbookServiceImpl.java

```
package com.ruoyi.ptbookManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.ptbookManager.mapper.PtbookMapper;
```

```
import com.ruoyi.ptbookManager.domain.Ptbook;
import com.ruoyi.ptbookManager.service.IPtbookService;

/**
 * 体测预约管理 Service 业务层处理
 *
 * @author LJH
 * @date 2024-11-26
 */
@Service
public class PtbookServiceImpl implements IPtbookService
{
    @Autowired
    private PtbookMapper ptbookMapper;

    /**
     * 查询体测预约管理
     *
     * @param ptbookId 体测预约管理主键
     * @return 体测预约管理
     */
    @Override
    public Ptbook selectPtbookByPtbookId(Long ptbookId)
    {
        return ptbookMapper.selectPtbookByPtbookId(ptbookId);
    }
```

```
/**
 * 查询体测预约管理列表
 *
 * @param ptbook 体测预约管理
 * @return 体测预约管理
 */
@Override
public List<Ptbook> selectPtbookList(Ptbook ptbook)
{
    return ptbookMapper.selectPtbookList(ptbook);
}

/**
 * 新增体测预约管理
 *
 * @param ptbook 体测预约管理
 * @return 结果
 */
@Override
public int insertPtbook(Ptbook ptbook)
{
    return ptbookMapper.insertPtbook(ptbook);
}

/**
 * 修改体测预约管理
 *
```

```
     * @param ptbook 体测预约管理
     * @return 结果
     */
    @Override
    public int updatePtbook(Ptbook ptbook)
    {
        return ptbookMapper.updatePtbook(ptbook);
    }

    /**
     * 批量删除体测预约管理
     *
     * @param ptbookIds 需要删除的体测预约管理主键
     * @return 结果
     */
    @Override
    public int deletePtbookByPtbookIds(Long[] ptbookIds)
    {
        return ptbookMapper.deletePtbookByPtbookIds(ptbookIds);
    }

    /**
     * 删除体测预约管理信息
     *
     * @param ptbookId 体测预约管理主键
     * @return 结果
     */
```

```
    @Override
    public int deletePtbookByPtbookId(Long ptbookId)
    {
        return ptbookMapper.deletePtbookByPtbookId(ptbookId);
    }
}
```

src/main/java/com/iot/ptManager/controller/PtstandController.java

```
package com.ruoyi.ptManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.ptManager.domain.Ptstand;
```

```
import com.ruoyi.ptManager.service.IPtstandService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
 * 体测标准 Controller
 *
 * @author ljh
 * @date 2024-10-30
 */
@RestController
@RequestMapping("/ptstandManager/ptstandManager")
public class PtstandController extends BaseController
{
    @Autowired
    private IPtstandService ptstandService;

    /**
     * 查询体测标准列表
     */
    @PreAuthorize("@ss.hasPermi('ptstandManager:ptstandManager:list')")
    @GetMapping("/list")
    public TableDataInfo list(Ptstand ptstand)
    {
        startPage();
        List<Ptstand> list = ptstandService.selectPtstandList(ptstand);
        return getDataTable(list);
```

```
}

/**
 * 导出体测标准列表
 */
@PreAuthorize("@ss.hasPermi('ptstandManager:ptstandManager:export')")
@Log(title = "体测标准", businessType = BusinessType.EXPORT)
@PostMapping("/export")
public void export(HttpServletResponse response, Ptstand ptstand)
{
    List<Ptstand> list = ptstandService.selectPtstandList(ptstand);
    ExcelUtil<Ptstand> util = new ExcelUtil<Ptstand>(Ptstand.class);
    util.exportExcel(response, list, "体测标准数据");
}

/**
 * 获取体测标准详细信息
 */
@PreAuthorize("@ss.hasPermi('ptstandManager:ptstandManager:query')")
@GetMapping(value = "/{standId}")
public AjaxResult getInfo(@PathVariable("standId") Long standId)
{
    return success(ptstandService.selectPtstandByStandId(standId));
}

/**
 * 新增体测标准
```

```
 */
@PreAuthorize("@ss.hasPermi('ptstandManager:ptstandManager:add')")
@Log(title = "体测标准", businessType = BusinessType.INSERT)
@PostMapping
public AjaxResult add(@RequestBody Ptstand ptstand)
{
    return toAjax(ptstandService.insertPtstand(ptstand));
}

/**
 * 修改体测标准
 */
@PreAuthorize("@ss.hasPermi('ptstandManager:ptstandManager:edit')")
@Log(title = "体测标准", businessType = BusinessType.UPDATE)
@PutMapping
public AjaxResult edit(@RequestBody Ptstand ptstand)
{
    return toAjax(ptstandService.updatePtstand(ptstand));
}

/**
 * 删除体测标准
 */
@PreAuthorize("@ss.hasPermi('ptstandManager:ptstandManager:remove')")
@Log(title = "体测标准", businessType = BusinessType.DELETE)
@DeleteMapping("/{standIds}")
public AjaxResult remove(@PathVariable Long[] standIds)
```

```
    {
        return toAjax(ptstandService.deletePtstandByStandIds(standIds));
    }
}
```

src/main/java/com/iot/ptManager/domain/Ptstand.java

```
package com.ruoyi.ptManager.domain;

import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 体测标准对象 ptstand
 *
 * @author ljh
 * @date 2024-10-30
 */
public class Ptstand extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 体测标准 id */
    private Long standId;
```

```
/** 体测项目 id */
@Excel(name = "体测项目 id")
private Long ptId;

/** 体测标准编号 */
@Excel(name = "体测标准编号")
private String standNum;

/** 标准等级 */
@Excel(name = "标准等级")
private String standLevel;

/** 得分 */
@Excel(name = "得分")
private Long standScore;

/** 年级 */
@Excel(name = "年级")
private String standGrade;

/** 性别 */
@Excel(name = "性别")
private String standGender;

/** 最低成绩 */
@Excel(name = "最低成绩")
private Long standMin;
```

```
/** 最高成绩 */
@Excel(name = "最高成绩")
private Long standMax;

public void setStandId(Long standId)
{
    this.standId = standId;
}

public Long getStandId()
{
    return standId;
}
public void setPtId(Long ptId)
{
    this.ptId = ptId;
}

public Long getPtId()
{
    return ptId;
}
public void setStandNum(String standNum)
{
    this.standNum = standNum;
}
```

```
public String getStandNum()
{
    return standNum;
}
public void setStandLevel(String standLevel)
{
    this.standLevel = standLevel;
}

public String getStandLevel()
{
    return standLevel;
}
public void setStandScore(Long standScore)
{
    this.standScore = standScore;
}

public Long getStandScore()
{
    return standScore;
}
public void setStandGrade(String standGrade)
{
    this.standGrade = standGrade;
}
```

```
public String getStandGrade()
{
    return standGrade;
}
public void setStandGender(String standGender)
{
    this.standGender = standGender;
}

public String getStandGender()
{
    return standGender;
}
public void setStandMin(Long standMin)
{
    this.standMin = standMin;
}

public Long getStandMin()
{
    return standMin;
}
public void setStandMax(Long standMax)
{
    this.standMax = standMax;
}
```

```
    public Long getStandMax()
    {
        return standMax;
    }

    @Override
    public String toString() {
        return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
            .append("standId", getStandId())
            .append("ptId", getPtId())
            .append("standNum", getStandNum())
            .append("standLevel", getStandLevel())
            .append("standScore", getStandScore())
            .append("standGrade", getStandGrade())
            .append("standGender", getStandGender())
            .append("standMin", getStandMin())
            .append("standMax", getStandMax())
            .toString();
    }
}
```

src/main/java/com/iot/ptManager/mapper/PtstandMapper.java

```
package com.ruoyi.ptManager.mapper;

import java.util.List;
```

```
import com.ruoyi.ptManager.domain.Ptstand;

/**
 * 体测标准 Mapper 接口
 *
 * @author ljh
 * @date 2024-10-30
 */
public interface PtstandMapper
{
    /**
     * 查询体测标准
     *
     * @param standId 体测标准主键
     * @return 体测标准
     */
    public Ptstand selectPtstandByStandId(Long standId);

    /**
     * 查询体测标准列表
     *
     * @param ptstand 体测标准
     * @return 体测标准集合
     */
    public List<Ptstand> selectPtstandList(Ptstand ptstand);

    /**
```

```
     * 新增体测标准
     *
     * @param ptstand 体测标准
     * @return 结果
     */
    public int insertPtstand(Ptstand ptstand);

    /**
     * 修改体测标准
     *
     * @param ptstand 体测标准
     * @return 结果
     */
    public int updatePtstand(Ptstand ptstand);

    /**
     * 删除体测标准
     *
     * @param standId 体测标准主键
     * @return 结果
     */
    public int deletePtstandByStandId(Long standId);

    /**
     * 批量删除体测标准
     *
     * @param standIds 需要删除的数据主键集合
```

```
     * @return  结果
     */
    public int deletePtstandByStandIds(Long[] standIds);
}
```

src/main/java/com/iot/ptManager/service/IPtstandService.java

```
package com.ruoyi.ptManager.service;

import java.util.List;
import com.ruoyi.ptManager.domain.Ptstand;

/**
 * 体测标准 Service 接口
 *
 * @author ljh
 * @date 2024-10-30
 */
public interface IPtstandService
{
    /**
     * 查询体测标准
     *
     * @param standId  体测标准主键
     * @return  体测标准
     */
    public Ptstand selectPtstandByStandId(Long standId);
```

```
/**
 * 查询体测标准列表
 *
 * @param ptstand 体测标准
 * @return 体测标准集合
 */
public List<Ptstand> selectPtstandList(Ptstand ptstand);

/**
 * 新增体测标准
 *
 * @param ptstand 体测标准
 * @return 结果
 */
public int insertPtstand(Ptstand ptstand);

/**
 * 修改体测标准
 *
 * @param ptstand 体测标准
 * @return 结果
 */
public int updatePtstand(Ptstand ptstand);

/**
 * 批量删除体测标准
```

```
     *
     * @param standIds 需要删除的体测标准主键集合
     * @return 结果
     */
    public int deletePtstandByStandIds(Long[] standIds);

    /**
     * 删除体测标准信息
     *
     * @param standId 体测标准主键
     * @return 结果
     */
    public int deletePtstandByStandId(Long standId);
}
```

src/main/java/com/iot/ptManager/service/impl/PtstandServiceImpl.java

```
package com.ruoyi.ptManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.ptManager.mapper.PtstandMapper;
import com.ruoyi.ptManager.domain.Ptstand;
import com.ruoyi.ptManager.service.IPtstandService;

/**
```

```
 * 体测标准 Service 业务层处理
 *
 * @author ljh
 * @date 2024-10-30
 */
@Service
public class PtstandServiceImpl implements IPtstandService
{
    @Autowired
    private PtstandMapper ptstandMapper;

    /**
     * 查询体测标准
     *
     * @param standId 体测标准主键
     * @return 体测标准
     */
    @Override
    public Ptstand selectPtstandByStandId(Long standId)
    {
        return ptstandMapper.selectPtstandByStandId(standId);
    }

    /**
     * 查询体测标准列表
     *
     * @param ptstand 体测标准
```

```
     * @return  体测标准
     */
    @Override
    public List<Ptstand> selectPtstandList(Ptstand ptstand)
    {
        return ptstandMapper.selectPtstandList(ptstand);
    }

    /**
     * 新增体测标准
     *
     * @param ptstand  体测标准
     * @return  结果
     */
    @Override
    public int insertPtstand(Ptstand ptstand)
    {
        return ptstandMapper.insertPtstand(ptstand);
    }

    /**
     * 修改体测标准
     *
     * @param ptstand  体测标准
     * @return  结果
     */
    @Override
```

```
public int updatePtstand(Ptstand ptstand)
{
    return ptstandMapper.updatePtstand(ptstand);
}

/**
 * 批量删除体测标准
 *
 * @param standIds 需要删除的体测标准主键
 * @return 结果
 */
@Override
public int deletePtstandByStandIds(Long[] standIds)
{
    return ptstandMapper.deletePtstandByStandIds(standIds);
}

/**
 * 删除体测标准信息
 *
 * @param standId 体测标准主键
 * @return 结果
 */
@Override
public int deletePtstandByStandId(Long standId)
{
    return ptstandMapper.deletePtstandByStandId(standId);
```

```
    }
}
```

src/main/java/com/iot/ptrecordManager/controller/PtrecordController.java

```
package com.ruoyi.ptrecordManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.ptrecordManager.domain.Ptrecord;
import com.ruoyi.ptrecordManager.service.IPtrecordService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;
```

```
/**
 * 体测记录管理 Controller
 *
 * @author LJH
 * @date 2024-11-26
 */
@RestController
@RequestMapping("/ptrecordManager/ptrecord")
public class PtrecordController extends BaseController
{
    @Autowired
    private IPtrecordService ptrecordService;

    /**
     * 查询体测记录管理列表
     */
    @PreAuthorize("@ss.hasPermi('ptrecordManager:ptrecord:list')")
    @GetMapping("/list")
    public TableDataInfo list(Ptrecord ptrecord)
    {
        startPage();
        List<Ptrecord> list = ptrecordService.selectPtrecordList(ptrecord);
        return getDataTable(list);
    }

    /**
     * 导出体测记录管理列表
```

```
 */
@PreAuthorize("@ss.hasPermi('ptrecordManager:ptrecord:export')")
@Log(title = "体测记录管理", businessType = BusinessType.EXPORT)
@PostMapping("/export")
public void export(HttpServletResponse response, Ptrecord ptrecord)
{
    List<Ptrecord> list = ptrecordService.selectPtrecordList(ptrecord);
    ExcelUtil<Ptrecord> util = new ExcelUtil<Ptrecord>(Ptrecord.class);
    util.exportExcel(response, list, "体测记录管理数据");
}

/**
 * 获取体测记录管理详细信息
 */
@PreAuthorize("@ss.hasPermi('ptrecordManager:ptrecord:query')")
@GetMapping(value = "/{ptrId}")
public AjaxResult getInfo(@PathVariable("ptrId") Long ptrId)
{
    return success(ptrecordService.selectPtrecordByPtrId(ptrId));
}

/**
 * 新增体测记录管理
 */
@PreAuthorize("@ss.hasPermi('ptrecordManager:ptrecord:add')")
@Log(title = "体测记录管理", businessType = BusinessType.INSERT)
@PostMapping
```

```
    public AjaxResult add(@RequestBody Ptrecord ptrecord)
    {
        return toAjax(ptrecordService.insertPtrecord(ptrecord));
    }

    /**
     * 修改体测记录管理
     */
    @PreAuthorize("@ss.hasPermi('ptrecordManager:ptrecord:edit')")
    @Log(title = "体测记录管理", businessType = BusinessType.UPDATE)
    @PutMapping
    public AjaxResult edit(@RequestBody Ptrecord ptrecord)
    {
        return toAjax(ptrecordService.updatePtrecord(ptrecord));
    }

    /**
     * 删除体测记录管理
     */
    @PreAuthorize("@ss.hasPermi('ptrecordManager:ptrecord:remove')")
    @Log(title = "体测记录管理", businessType = BusinessType.DELETE)
    @DeleteMapping("/{ptrIds}")
    public AjaxResult remove(@PathVariable Long[] ptrIds)
    {
        return toAjax(ptrecordService.deletePtrecordByPtrIds(ptrIds));
    }
}
```

src/main/java/com/iot/ptrecordManager/domain/Ptrecord.java

```
package com.ruoyi.ptrecordManager.domain;

import java.util.Date;
import com.fasterxml.jackson.annotation.JsonFormat;
import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 体测记录管理对象 ptrecord
 *
 * @author LJH
 * @date 2024-11-26
 */
public class Ptrecord extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 记录 id */
    private Long ptrId;

    /** 机器 id */
    @Excel(name = "机器 id")
```

```
private Long macId;

/** 管理员 id */
@Excel(name = "管理员 id")
private Long adminId;

/** 项目 id */
@Excel(name = "项目 id")
private Long ptId;

/** 学生 id */
@Excel(name = "学生 id")
private Long stuId;

/** 记录序号 */
@Excel(name = "记录序号")
private String ptrNum;

/** 成绩 */
@Excel(name = "成绩")
private String ptrScore;

/** 权重分 */
@Excel(name = "权重分")
private Long ptrWscore;

/** 测试日期 */
```

```
@JsonFormat(pattern = "yyyy-MM-dd")
@Excel(name = "测试日期", width = 30, dateFormat = "yyyy-MM-dd")
private Date ptrDate;

public void setPtrId(Long ptrId)
{
    this.ptrId = ptrId;
}

public Long getPtrId()
{
    return ptrId;
}
public void setMacId(Long macId)
{
    this.macId = macId;
}

public Long getMacId()
{
    return macId;
}
public void setAdminId(Long adminId)
{
    this.adminId = adminId;
}
```

```
public Long getAdminId()
{
    return adminId;
}
public void setPtId(Long ptId)
{
    this.ptId = ptId;
}

public Long getPtId()
{
    return ptId;
}
public void setStuId(Long stuId)
{
    this.stuId = stuId;
}

public Long getStuId()
{
    return stuId;
}
public void setPtrNum(String ptrNum)
{
    this.ptrNum = ptrNum;
}
```

```
public String getPtrNum()
{
    return ptrNum;
}
public void setPtrScore(String ptrScore)
{
    this.ptrScore = ptrScore;
}

public String getPtrScore()
{
    return ptrScore;
}
public void setPtrWscore(Long ptrWscore)
{
    this.ptrWscore = ptrWscore;
}

public Long getPtrWscore()
{
    return ptrWscore;
}
public void setPtrDate(Date ptrDate)
{
    this.ptrDate = ptrDate;
}
```

```
    public Date getPtrDate()
    {
        return ptrDate;
    }

    @Override
    public String toString() {
        return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
            .append("ptrId", getPtrId())
            .append("macId", getMacId())
            .append("adminId", getAdminId())
            .append("ptId", getPtId())
            .append("stuId", getStuId())
            .append("ptrNum", getPtrNum())
            .append("ptrScore", getPtrScore())
            .append("ptrWscore", getPtrWscore())
            .append("ptrDate", getPtrDate())
            .toString();
    }
}
```

src/main/java/com/iot/ptrecordManager/mapper/PtrecordMapper.java

```
package com.ruoyi.ptrecordManager.mapper;

import java.util.List;
import com.ruoyi.ptrecordManager.domain.Ptrecord;
```

```
/**
 * 体测记录管理 Mapper 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface PtrecordMapper
{
    /**
     * 查询体测记录管理
     *
     * @param ptrId 体测记录管理主键
     * @return 体测记录管理
     */
    public Ptrecord selectPtrecordByPtrId(Long ptrId);

    /**
     * 查询体测记录管理列表
     *
     * @param ptrecord 体测记录管理
     * @return 体测记录管理集合
     */
    public List<Ptrecord> selectPtrecordList(Ptrecord ptrecord);

    /**
     * 新增体测记录管理
```

```
 *
 * @param ptrecord 体测记录管理
 * @return 结果
 */
public int insertPtrecord(Ptrecord ptrecord);

/**
 * 修改体测记录管理
 *
 * @param ptrecord 体测记录管理
 * @return 结果
 */
public int updatePtrecord(Ptrecord ptrecord);

/**
 * 删除体测记录管理
 *
 * @param ptrId 体测记录管理主键
 * @return 结果
 */
public int deletePtrecordByPtrId(Long ptrId);

/**
 * 批量删除体测记录管理
 *
 * @param ptrIds 需要删除的数据主键集合
 * @return 结果
```

```
     */
    public int deletePtrecordByPtrIds(Long[] ptrIds);
}
```

src/main/java/com/iot/ptrecordManager/service/IPtrecordService.java

```
package com.ruoyi.ptrecordManager.service;

import java.util.List;
import com.ruoyi.ptrecordManager.domain.Ptrecord;

/**
 * 体测记录管理 Service 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface IPtrecordService
{
    /**
     * 查询体测记录管理
     *
     * @param ptrId 体测记录管理主键
     * @return 体测记录管理
     */
    public Ptrecord selectPtrecordByPtrId(Long ptrId);
```

```
/**
 * 查询体测记录管理列表
 *
 * @param ptrecord 体测记录管理
 * @return 体测记录管理集合
 */
public List<Ptrecord> selectPtrecordList(Ptrecord ptrecord);

/**
 * 新增体测记录管理
 *
 * @param ptrecord 体测记录管理
 * @return 结果
 */
public int insertPtrecord(Ptrecord ptrecord);

/**
 * 修改体测记录管理
 *
 * @param ptrecord 体测记录管理
 * @return 结果
 */
public int updatePtrecord(Ptrecord ptrecord);

/**
 * 批量删除体测记录管理
 *
```

```
     * @param ptrIds 需要删除的体测记录管理主键集合
     * @return 结果
     */
    public int deletePtrecordByPtrIds(Long[] ptrIds);

    /**
     * 删除体测记录管理信息
     *
     * @param ptrId 体测记录管理主键
     * @return 结果
     */
    public int deletePtrecordByPtrId(Long ptrId);
}
```

src/main/java/com/iot/ptrecordManager/service/impl/PtrecordServiceImpl.java

```
package com.ruoyi.ptrecordManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.ptrecordManager.mapper.PtrecordMapper;
import com.ruoyi.ptrecordManager.domain.Ptrecord;
import com.ruoyi.ptrecordManager.service.IPtrecordService;

/**
 * 体测记录管理 Service 业务层处理
```

```
 *
 * @author LJH
 * @date 2024-11-26
 */
@Service
public class PtrecordServiceImpl implements IPtrecordService
{
    @Autowired
    private PtrecordMapper ptrecordMapper;

    /**
     * 查询体测记录管理
     *
     * @param ptrId 体测记录管理主键
     * @return 体测记录管理
     */
    @Override
    public Ptrecord selectPtrecordByPtrId(Long ptrId)
    {
        return ptrecordMapper.selectPtrecordByPtrId(ptrId);
    }

    /**
     * 查询体测记录管理列表
     *
     * @param ptrecord 体测记录管理
     * @return 体测记录管理
```

```
     */
    @Override
    public List<Ptrecord> selectPtrecordList(Ptrecord ptrecord)
    {
        return ptrecordMapper.selectPtrecordList(ptrecord);
    }

    /**
     * 新增体测记录管理
     *
     * @param ptrecord 体测记录管理
     * @return 结果
     */
    @Override
    public int insertPtrecord(Ptrecord ptrecord)
    {
        return ptrecordMapper.insertPtrecord(ptrecord);
    }

    /**
     * 修改体测记录管理
     *
     * @param ptrecord 体测记录管理
     * @return 结果
     */
    @Override
    public int updatePtrecord(Ptrecord ptrecord)
```

```
{
        return ptrecordMapper.updatePtrecord(ptrecord);
}

/**
 * 批量删除体测记录管理
 *
 * @param ptrIds 需要删除的体测记录管理主键
 * @return 结果
 */
@Override
public int deletePtrecordByPtrIds(Long[] ptrIds)
{
        return ptrecordMapper.deletePtrecordByPtrIds(ptrIds);
}

/**
 * 删除体测记录管理信息
 *
 * @param ptrId 体测记录管理主键
 * @return 结果
 */
@Override
public int deletePtrecordByPtrId(Long ptrId)
{
        return ptrecordMapper.deletePtrecordByPtrId(ptrId);
}
```

}

src/main/java/com/iot/ptscoreManager/controller/PtscoreController.java

```
package com.ruoyi.ptscoreManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.ptscoreManager.domain.Ptscore;
import com.ruoyi.ptscoreManager.service.IPtscoreService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
```

```
 * 学生分数管理 Controller
 *
 * @author LJH
 * @date 2024-11-26
 */
@RestController
@RequestMapping("/ptscoreManager/ptscore")
public class PtscoreController extends BaseController
{
    @Autowired
    private IPtscoreService ptscoreService;

    /**
     * 查询学生分数管理列表
     */
    @PreAuthorize("@ss.hasPermi('ptscoreManager:ptscore:list')")
    @GetMapping("/list")
    public TableDataInfo list(Ptscore ptscore)
    {
        startPage();
        List<Ptscore> list = ptscoreService.selectPtscoreList(ptscore);
        return getDataTable(list);
    }

    /**
     * 导出学生分数管理列表
     */
```

```
@PreAuthorize("@ss.hasPermi('ptscoreManager:ptscore:export')")
@Log(title = "学生分数管理", businessType = BusinessType.EXPORT)
@PostMapping("/export")
public void export(HttpServletResponse response, Ptscore ptscore)
{
    List<Ptscore> list = ptscoreService.selectPtscoreList(ptscore);
    ExcelUtil<Ptscore> util = new ExcelUtil<Ptscore>(Ptscore.class);
    util.exportExcel(response, list, "学生分数管理数据");
}

/**
 * 获取学生分数管理详细信息
 */
@PreAuthorize("@ss.hasPermi('ptscoreManager:ptscore:query')")
@GetMapping(value = "/{ptscoreId}")
public AjaxResult getInfo(@PathVariable("ptscoreId") Long ptscoreId)
{
    return success(ptscoreService.selectPtscoreByPtscoreId(ptscoreId));
}

/**
 * 新增学生分数管理
 */
@PreAuthorize("@ss.hasPermi('ptscoreManager:ptscore:add')")
@Log(title = "学生分数管理", businessType = BusinessType.INSERT)
@PostMapping
public AjaxResult add(@RequestBody Ptscore ptscore)
```

```
    {
        return toAjax(ptscoreService.insertPtscore(ptscore));
    }

    /**
     * 修改学生分数管理
     */
    @PreAuthorize("@ss.hasPermi('ptscoreManager:ptscore:edit')")
    @Log(title = "学生分数管理", businessType = BusinessType.UPDATE)
    @PutMapping
    public AjaxResult edit(@RequestBody Ptscore ptscore)
    {
        return toAjax(ptscoreService.updatePtscore(ptscore));
    }

    /**
     * 删除学生分数管理
     */
    @PreAuthorize("@ss.hasPermi('ptscoreManager:ptscore:remove')")
    @Log(title = "学生分数管理", businessType = BusinessType.DELETE)
    @DeleteMapping("/{ptscoreIds}")
    public AjaxResult remove(@PathVariable Long[] ptscoreIds)
    {
        return toAjax(ptscoreService.deletePtscoreByPtscoreIds(ptscoreIds));
    }
}
```

src/main/java/com/iot/ptscoreManager/domain/Ptscore.java

```
package com.ruoyi.ptscoreManager.domain;

import java.util.Date;
import com.fasterxml.jackson.annotation.JsonFormat;
import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 学生分数管理对象 ptscore
 *
 * @author LJH
 * @date 2024-11-26
 */
public class Ptscore extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 成绩 id */
    private Long ptscoreId;

    /** 学生 id */
    @Excel(name = "学生 id")
    private Long stuId;
```

```
/** 成绩 */
@Excel(name = "成绩")
private Long ptscoreScore;

/** 起始日期 */
@JsonFormat(pattern = "yyyy-MM-dd")
@Excel(name = "起始日期", width = 30, dateFormat = "yyyy-MM-dd")
private Date ptscoreSatart;

/** 结束日期 */
@JsonFormat(pattern = "yyyy-MM-dd")
@Excel(name = "结束日期", width = 30, dateFormat = "yyyy-MM-dd")
private Date ptscoreEnd;

/** 是否缺项 */
@Excel(name = "是否缺项")
private Integer ptscoreLack;

public void setPtscoreId(Long ptscoreId)
{
    this.ptscoreId = ptscoreId;
}

public Long getPtscoreId()
{
    return ptscoreId;
```

```
}
public void setStuId(Long stuId)
{
    this.stuId = stuId;
}

public Long getStuId()
{
    return stuId;
}
public void setPtscoreScore(Long ptscoreScore)
{
    this.ptscoreScore = ptscoreScore;
}

public Long getPtscoreScore()
{
    return ptscoreScore;
}
public void setPtscoreSatart(Date ptscoreSatart)
{
    this.ptscoreSatart = ptscoreSatart;
}

public Date getPtscoreSatart()
{
    return ptscoreSatart;
```

```
}
public void setPtscoreEnd(Date ptscoreEnd)
{
	this.ptscoreEnd = ptscoreEnd;
}

public Date getPtscoreEnd()
{
	return ptscoreEnd;
}
public void setPtscoreLack(Integer ptscoreLack)
{
	this.ptscoreLack = ptscoreLack;
}

public Integer getPtscoreLack()
{
	return ptscoreLack;
}

@Override
public String toString() {
	return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
		.append("ptscoreId", getPtscoreId())
		.append("stuId", getStuId())
		.append("ptscoreScore", getPtscoreScore())
		.append("ptscoreSatart", getPtscoreSatart())
```

```
                .append("ptscoreEnd", getPtscoreEnd())
                .append("ptscoreLack", getPtscoreLack())
                .toString();
    }
}
```

src/main/java/com/iot/ptscoreManager/mapper/PtscoreMapper.java

```
package com.ruoyi.ptscoreManager.mapper;

import java.util.List;
import com.ruoyi.ptscoreManager.domain.Ptscore;

/**
 * 学生分数管理 Mapper 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface PtscoreMapper
{
    /**
     * 查询学生分数管理
     *
     * @param ptscoreId 学生分数管理主键
     * @return 学生分数管理
     */
```

```
public Ptscore selectPtscoreByPtscoreId(Long ptscoreId);

/**
 * 查询学生分数管理列表
 *
 * @param ptscore 学生分数管理
 * @return 学生分数管理集合
 */
public List<Ptscore> selectPtscoreList(Ptscore ptscore);

/**
 * 新增学生分数管理
 *
 * @param ptscore 学生分数管理
 * @return 结果
 */
public int insertPtscore(Ptscore ptscore);

/**
 * 修改学生分数管理
 *
 * @param ptscore 学生分数管理
 * @return 结果
 */
public int updatePtscore(Ptscore ptscore);

/**
```

```
     * 删除学生分数管理
     *
     * @param ptscoreId 学生分数管理主键
     * @return 结果
     */
    public int deletePtscoreByPtscoreId(Long ptscoreId);

    /**
     * 批量删除学生分数管理
     *
     * @param ptscoreIds 需要删除的数据主键集合
     * @return 结果
     */
    public int deletePtscoreByPtscoreIds(Long[] ptscoreIds);
}
```

src/main/java/com/iot/ptscoreManager/service/IPtscoreService.java

```
package com.ruoyi.ptscoreManager.service;

import java.util.List;
import com.ruoyi.ptscoreManager.domain.Ptscore;

/**
 * 学生分数管理 Service 接口
 *
 * @author LJH
```

```
 * @date 2024-11-26
 */
public interface IPtscoreService
{
    /**
     * 查询学生分数管理
     *
     * @param ptscoreId 学生分数管理主键
     * @return 学生分数管理
     */
    public Ptscore selectPtscoreByPtscoreId(Long ptscoreId);

    /**
     * 查询学生分数管理列表
     *
     * @param ptscore 学生分数管理
     * @return 学生分数管理集合
     */
    public List<Ptscore> selectPtscoreList(Ptscore ptscore);

    /**
     * 新增学生分数管理
     *
     * @param ptscore 学生分数管理
     * @return 结果
     */
    public int insertPtscore(Ptscore ptscore);
```

```
    /**
     * 修改学生分数管理
     *
     * @param ptscore 学生分数管理
     * @return 结果
     */
    public int updatePtscore(Ptscore ptscore);

    /**
     * 批量删除学生分数管理
     *
     * @param ptscoreIds 需要删除的学生分数管理主键集合
     * @return 结果
     */
    public int deletePtscoreByPtscoreIds(Long[] ptscoreIds);

    /**
     * 删除学生分数管理信息
     *
     * @param ptscoreId 学生分数管理主键
     * @return 结果
     */
    public int deletePtscoreByPtscoreId(Long ptscoreId);
}
```

src/main/java/com/iot/ptscoreManager/service/impl/PtscoreServiceImpl.java

```
package com.ruoyi.ptscoreManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.ptscoreManager.mapper.PtscoreMapper;
import com.ruoyi.ptscoreManager.domain.Ptscore;
import com.ruoyi.ptscoreManager.service.IPtscoreService;

/**
 * 学生分数管理 Service 业务层处理
 *
 * @author LJH
 * @date 2024-11-26
 */
@Service
public class PtscoreServiceImpl implements IPtscoreService
{
    @Autowired
    private PtscoreMapper ptscoreMapper;

    /**
     * 查询学生分数管理
     *
     * @param ptscoreId 学生分数管理主键
     * @return 学生分数管理
```

```
     */
    @Override
    public Ptscore selectPtscoreByPtscoreId(Long ptscoreId)
    {
        return ptscoreMapper.selectPtscoreByPtscoreId(ptscoreId);
    }

    /**
     * 查询学生分数管理列表
     *
     * @param ptscore 学生分数管理
     * @return 学生分数管理
     */
    @Override
    public List<Ptscore> selectPtscoreList(Ptscore ptscore)
    {
        return ptscoreMapper.selectPtscoreList(ptscore);
    }

    /**
     * 新增学生分数管理
     *
     * @param ptscore 学生分数管理
     * @return 结果
     */
    @Override
    public int insertPtscore(Ptscore ptscore)
```

```
{
        return ptscoreMapper.insertPtscore(ptscore);
}

/**
 * 修改学生分数管理
 *
 * @param ptscore 学生分数管理
 * @return 结果
 */
@Override
public int updatePtscore(Ptscore ptscore)
{
        return ptscoreMapper.updatePtscore(ptscore);
}

/**
 * 批量删除学生分数管理
 *
 * @param ptscoreIds 需要删除的学生分数管理主键
 * @return 结果
 */
@Override
public int deletePtscoreByPtscoreIds(Long[] ptscoreIds)
{
        return ptscoreMapper.deletePtscoreByPtscoreIds(ptscoreIds);
}
```

```
    /**
     * 删除学生分数管理信息
     *
     * @param ptscoreId 学生分数管理主键
     * @return 结果
     */
    @Override
    public int deletePtscoreByPtscoreId(Long ptscoreId)
    {
        return ptscoreMapper.deletePtscoreByPtscoreId(ptscoreId);
    }
}
```

src/main/java/com/iot/stuManager/controller/StudentController.java

```
package com.ruoyi.stuManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.stuManager.domain.Student;
import com.ruoyi.stuManager.service.IStudentService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
 * 学生信息管理 Controller
 *
 * @author LJH
 * @date 2024-10-09
 */
@RestController
@RequestMapping("/stuManager/stu")
public class StudentController extends BaseController
{
    @Autowired
    private IStudentService studentService;

    /**
     * 查询学生信息管理列表
     */
```

```
@PreAuthorize("@ss.hasPermi('stuManager:stu:list')")
@GetMapping("/list")
public TableDataInfo list(Student student)
{
    startPage();
    List<Student> list = studentService.selectStudentList(student);
    return getDataTable(list);
}

/**
 * 导出学生信息管理列表
 */
@PreAuthorize("@ss.hasPermi('stuManager:stu:export')")
@Log(title = "学生信息管理", businessType = BusinessType.EXPORT)
@PostMapping("/export")
public void export(HttpServletResponse response, Student student)
{
    List<Student> list = studentService.selectStudentList(student);
    ExcelUtil<Student> util = new ExcelUtil<Student>(Student.class);
    util.exportExcel(response, list, "学生信息管理数据");
}

/**
 * 获取学生信息管理详细信息
 */
@PreAuthorize("@ss.hasPermi('stuManager:stu:query')")
@GetMapping(value = "/{stuId}")
```

```
public AjaxResult getInfo(@PathVariable("stuId") Long stuId)
{
    return success(studentService.selectStudentByStuId(stuId));
}

/**
 * 新增学生信息管理
 */
@PreAuthorize("@ss.hasPermi('stuManager:stu:add')")
@Log(title = "学生信息管理", businessType = BusinessType.INSERT)
@PostMapping
public AjaxResult add(@RequestBody Student student)
{
    return toAjax(studentService.insertStudent(student));
}

/**
 * 修改学生信息管理
 */
@PreAuthorize("@ss.hasPermi('stuManager:stu:edit')")
@Log(title = "学生信息管理", businessType = BusinessType.UPDATE)
@PutMapping
public AjaxResult edit(@RequestBody Student student)
{
    return toAjax(studentService.updateStudent(student));
}
```

```
    /**
     * 删除学生信息管理
     */
    @PreAuthorize("@ss.hasPermi('stuManager:stu:remove')")
    @Log(title = "学生信息管理", businessType = BusinessType.DELETE)
    @DeleteMapping("/{stuIds}")
    public AjaxResult remove(@PathVariable Long[] stuIds)
    {
        return toAjax(studentService.deleteStudentByStuIds(stuIds));
    }
}
```

src/main/java/com/iot/stuManager/domain/Student.java

```
package com.ruoyi.stuManager.domain;

import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 学生信息管理对象 student
 *
 * @author LJH
 * @date 2024-10-09
 */
```

```
public class Student extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 学生 id */
    private Long stuId;

    /** 学校 id */
    private Long schId;

    /** 学号 */
    @Excel(name = "学号")
    private String stuNum;

    /** 姓名 */
    @Excel(name = "姓名")
    private String stuName;

    /** 性别 */
    @Excel(name = "性别")
    private String stuSex;

    /** 学院 */
    @Excel(name = "学院")
    private String college;

    /** 专业 */
```

```
@Excel(name = "专业")
private String major;

/** 班级 */
@Excel(name = "班级")
private String stuClass;

/** 密码 */
private String stuPwd;

/** 电话 */
@Excel(name = "电话")
private String stuPhone;

/** 电子邮件 */
@Excel(name = "电子邮件")
private String stuEmail;

/** 头像 */
@Excel(name = "头像")
private String stuAvatar;

public void setStuId(Long stuId)
{
    this.stuId = stuId;
}
```

```
public Long getStuId()
{
    return stuId;
}
public void setSchId(Long schId)
{
    this.schId = schId;
}

public Long getSchId()
{
    return schId;
}
public void setStuNum(String stuNum)
{
    this.stuNum = stuNum;
}

public String getStuNum()
{
    return stuNum;
}
public void setStuName(String stuName)
{
    this.stuName = stuName;
}
```

```
public String getStuName()
{
    return stuName;
}
public void setStuSex(String stuSex)
{
    this.stuSex = stuSex;
}

public String getStuSex()
{
    return stuSex;
}
public void setCollege(String college)
{
    this.college = college;
}

public String getCollege()
{
    return college;
}
public void setMajor(String major)
{
    this.major = major;
}
```

```
public String getMajor()
{
    return major;
}
public void setStuClass(String stuClass)
{
    this.stuClass = stuClass;
}

public String getStuClass()
{
    return stuClass;
}
public void setStuPwd(String stuPwd)
{
    this.stuPwd = stuPwd;
}

public String getStuPwd()
{
    return stuPwd;
}
public void setStuPhone(String stuPhone)
{
    this.stuPhone = stuPhone;
}
```

```
public String getStuPhone()
{
	return stuPhone;
}
public void setStuEmail(String stuEmail)
{
	this.stuEmail = stuEmail;
}

public String getStuEmail()
{
	return stuEmail;
}
public void setStuAvatar(String stuAvatar)
{
	this.stuAvatar = stuAvatar;
}

public String getStuAvatar()
{
	return stuAvatar;
}

@Override
public String toString() {
	return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
		.append("stuId", getStuId())
```

```
                .append("schId", getSchId())
                .append("stuNum", getStuNum())
                .append("stuName", getStuName())
                .append("stuSex", getStuSex())
                .append("college", getCollege())
                .append("major", getMajor())
                .append("stuClass", getStuClass())
                .append("stuPwd", getStuPwd())
                .append("stuPhone", getStuPhone())
                .append("stuEmail", getStuEmail())
                .append("stuAvatar", getStuAvatar())
                .toString();
    }
}
```

src/main/java/com/iot/stuManager/mapper/StudentMapper.java

```
package com.ruoyi.stuManager.mapper;

import java.util.List;
import com.ruoyi.stuManager.domain.Student;

/**
 * 学生信息管理 Mapper 接口
 *
 * @author LJH
 * @date 2024-10-09
```

```
 */
public interface StudentMapper
{
    /**
     * 查询学生信息管理
     *
     * @param stuId 学生信息管理主键
     * @return 学生信息管理
     */
    public Student selectStudentByStuId(Long stuId);

    /**
     * 查询学生信息管理列表
     *
     * @param student 学生信息管理
     * @return 学生信息管理集合
     */
    public List<Student> selectStudentList(Student student);

    /**
     * 新增学生信息管理
     *
     * @param student 学生信息管理
     * @return 结果
     */
    public int insertStudent(Student student);
```

```
    /**
     * 修改学生信息管理
     *
     * @param student 学生信息管理
     * @return 结果
     */
    public int updateStudent(Student student);

    /**
     * 删除学生信息管理
     *
     * @param stuId 学生信息管理主键
     * @return 结果
     */
    public int deleteStudentByStuId(Long stuId);

    /**
     * 批量删除学生信息管理
     *
     * @param stuIds 需要删除的数据主键集合
     * @return 结果
     */
    public int deleteStudentByStuIds(Long[] stuIds);
}
```

src/main/java/com/iot/stuManager/service/IStudentService.java

```
package com.ruoyi.stuManager.service;

import java.util.List;
import com.ruoyi.stuManager.domain.Student;

/**
 * 学生信息管理 Service 接口
 *
 * @author LJH
 * @date 2024-10-09
 */
public interface IStudentService
{
    /**
     * 查询学生信息管理
     *
     * @param stuId 学生信息管理主键
     * @return 学生信息管理
     */
    public Student selectStudentByStuId(Long stuId);

    /**
     * 查询学生信息管理列表
     *
     * @param student 学生信息管理
     * @return 学生信息管理集合
     */
```

```
public List<Student> selectStudentList(Student student);

/**
 * 新增学生信息管理
 *
 * @param student 学生信息管理
 * @return 结果
 */
public int insertStudent(Student student);

/**
 * 修改学生信息管理
 *
 * @param student 学生信息管理
 * @return 结果
 */
public int updateStudent(Student student);

/**
 * 批量删除学生信息管理
 *
 * @param stuIds 需要删除的学生信息管理主键集合
 * @return 结果
 */
public int deleteStudentByStuIds(Long[] stuIds);

/**
```

```
     * 删除学生信息管理信息
     *
     * @param stuId 学生信息管理主键
     * @return 结果
     */
    public int deleteStudentByStuId(Long stuId);
}
```

src/main/java/com/iot/stuManager/service/impl/StudentServiceImpl.java

```
package com.ruoyi.stuManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.stuManager.mapper.StudentMapper;
import com.ruoyi.stuManager.domain.Student;
import com.ruoyi.stuManager.service.IStudentService;

/**
 * 学生信息管理 Service 业务层处理
 *
 * @author LJH
 * @date 2024-10-09
 */
@Service
public class StudentServiceImpl implements IStudentService
```

```
{
    @Autowired
    private StudentMapper studentMapper;

    /**
     * 查询学生信息管理
     *
     * @param stuId 学生信息管理主键
     * @return 学生信息管理
     */
    @Override
    public Student selectStudentByStuId(Long stuId)
    {
        return studentMapper.selectStudentByStuId(stuId);
    }

    /**
     * 查询学生信息管理列表
     *
     * @param student 学生信息管理
     * @return 学生信息管理
     */
    @Override
    public List<Student> selectStudentList(Student student)
    {
        return studentMapper.selectStudentList(student);
    }
```

```
/**
 * 新增学生信息管理
 *
 * @param student 学生信息管理
 * @return 结果
 */
@Override
public int insertStudent(Student student)
{
    return studentMapper.insertStudent(student);
}

/**
 * 修改学生信息管理
 *
 * @param student 学生信息管理
 * @return 结果
 */
@Override
public int updateStudent(Student student)
{
    return studentMapper.updateStudent(student);
}

/**
 * 批量删除学生信息管理
```

```
     *
     * @param stuIds 需要删除的学生信息管理主键
     * @return 结果
     */
    @Override
    public int deleteStudentByStuIds(Long[] stuIds)
    {
        return studentMapper.deleteStudentByStuIds(stuIds);
    }

    /**
     * 删除学生信息管理信息
     *
     * @param stuId 学生信息管理主键
     * @return 结果
     */
    @Override
    public int deleteStudentByStuId(Long stuId)
    {
        return studentMapper.deleteStudentByStuId(stuId);
    }
}
```

src/main/java/com/iot/teaManager/controller/TeacherController.java

```
package com.ruoyi.teaManager.controller;

import java.util.List;
```

```
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.teaManager.domain.Teacher;
import com.ruoyi.teaManager.service.ITeacherService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
 * 教师 Controller
 *
 * @author ljh
 * @date 2024-10-30
 */
@RestController
```

```
@RequestMapping("/teaManager/teaManager")
public class TeacherController extends BaseController
{
    @Autowired
    private ITeacherService teacherService;

    /**
     * 查询教师列表
     */
    @PreAuthorize("@ss.hasPermi('teaManager:teaManager:list')")
    @GetMapping("/list")
    public TableDataInfo list(Teacher teacher)
    {
        startPage();
        List<Teacher> list = teacherService.selectTeacherList(teacher);
        return getDataTable(list);
    }

    /**
     * 导出教师列表
     */
    @PreAuthorize("@ss.hasPermi('teaManager:teaManager:export')")
    @Log(title = "教师", businessType = BusinessType.EXPORT)
    @PostMapping("/export")
    public void export(HttpServletResponse response, Teacher teacher)
    {
        List<Teacher> list = teacherService.selectTeacherList(teacher);
```

```
        ExcelUtil<Teacher> util = new ExcelUtil<Teacher>(Teacher.class);
        util.exportExcel(response, list, "教师数据");
    }

    /**
     * 获取教师详细信息
     */
    @PreAuthorize("@ss.hasPermi('teaManager:teaManager:query')")
    @GetMapping(value = "/{teaId}")
    public AjaxResult getInfo(@PathVariable("teaId") Long teaId)
    {
        return success(teacherService.selectTeacherByTeaId(teaId));
    }

    /**
     * 新增教师
     */
    @PreAuthorize("@ss.hasPermi('teaManager:teaManager:add')")
    @Log(title = "教师", businessType = BusinessType.INSERT)
    @PostMapping
    public AjaxResult add(@RequestBody Teacher teacher)
    {
        return toAjax(teacherService.insertTeacher(teacher));
    }

    /**
     * 修改教师
```

```
     */
    @PreAuthorize("@ss.hasPermi('teaManager:teaManager:edit')")
    @Log(title = "教师", businessType = BusinessType.UPDATE)
    @PutMapping
    public AjaxResult edit(@RequestBody Teacher teacher)
    {
        return toAjax(teacherService.updateTeacher(teacher));
    }

    /**
     * 删除教师
     */
    @PreAuthorize("@ss.hasPermi('teaManager:teaManager:remove')")
    @Log(title = "教师", businessType = BusinessType.DELETE)
    @DeleteMapping("/{teaIds}")
    public AjaxResult remove(@PathVariable Long[] teaIds)
    {
        return toAjax(teacherService.deleteTeacherByTeaIds(teaIds));
    }
}
```

src/main/java/com/iot/teaManager/domain/Teacher.java

```
package com.ruoyi.teaManager.domain;

import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
```

```
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 教师对象 teacher
 *
 * @author ljh
 * @date 2024-10-30
 */
public class Teacher extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 教师 id */
    private Long teaId;

    /** 教师所在学校 id */
    private Long schId;

    /** 教师工号 */
    @Excel(name = "教师工号")
    private String teaNum;

    /** 教师名称 */
    @Excel(name = "教师名称")
    private String teaName;
```

```
/** 教师性别 */
@Excel(name = "教师性别")
private String teaSex;

/** 教师密码 */
private String teaPwd;

/** 教师电话 */
@Excel(name = "教师电话")
private String teaPhone;

/** 教师邮箱 */
@Excel(name = "教师邮箱")
private String teaEmail;

/** 教师头像 */
@Excel(name = "教师头像")
private String teaAvatar;

public void setTeaId(Long teaId)
{
    this.teaId = teaId;
}

public Long getTeaId()
{
    return teaId;
```

```
}
public void setSchId(Long schId)
{
	this.schId = schId;
}

public Long getSchId()
{
	return schId;
}
public void setTeaNum(String teaNum)
{
	this.teaNum = teaNum;
}

public String getTeaNum()
{
	return teaNum;
}
public void setTeaName(String teaName)
{
	this.teaName = teaName;
}

public String getTeaName()
{
	return teaName;
```

```
}
public void setTeaSex(String teaSex)
{
	this.teaSex = teaSex;
}

public String getTeaSex()
{
	return teaSex;
}
public void setTeaPwd(String teaPwd)
{
	this.teaPwd = teaPwd;
}

public String getTeaPwd()
{
	return teaPwd;
}
public void setTeaPhone(String teaPhone)
{
	this.teaPhone = teaPhone;
}

public String getTeaPhone()
{
	return teaPhone;
```

```
}
public void setTeaEmail(String teaEmail)
{
	this.teaEmail = teaEmail;
}

public String getTeaEmail()
{
	return teaEmail;
}
public void setTeaAvatar(String teaAvatar)
{
	this.teaAvatar = teaAvatar;
}

public String getTeaAvatar()
{
	return teaAvatar;
}

@Override
public String toString() {
	return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
		.append("teaId", getTeaId())
		.append("schId", getSchId())
		.append("teaNum", getTeaNum())
		.append("teaName", getTeaName())
```

```
                .append("teaSex", getTeaSex())
                .append("teaPwd", getTeaPwd())
                .append("teaPhone", getTeaPhone())
                .append("teaEmail", getTeaEmail())
                .append("teaAvatar", getTeaAvatar())
                .toString();
    }
}
```

src/main/java/com/iot/teaManager/mapper/TeacherMapper.java

```
package com.ruoyi.teaManager.mapper;

import java.util.List;
import com.ruoyi.teaManager.domain.Teacher;

/**
 * 教师 Mapper 接口
 *
 * @author ljh
 * @date 2024-10-30
 */
public interface TeacherMapper
{
    /**
     * 查询教师
     *
```

```
     * @param teaId  教师主键
     * @return  教师
     */
    public Teacher selectTeacherByTeaId(Long teaId);

    /**
     * 查询教师列表
     *
     * @param teacher  教师
     * @return  教师集合
     */
    public List<Teacher> selectTeacherList(Teacher teacher);

    /**
     * 新增教师
     *
     * @param teacher  教师
     * @return  结果
     */
    public int insertTeacher(Teacher teacher);

    /**
     * 修改教师
     *
     * @param teacher  教师
     * @return  结果
     */
```

```
    public int updateTeacher(Teacher teacher);

    /**
     * 删除教师
     *
     * @param teaId 教师主键
     * @return 结果
     */
    public int deleteTeacherByTeaId(Long teaId);

    /**
     * 批量删除教师
     *
     * @param teaIds 需要删除的数据主键集合
     * @return 结果
     */
    public int deleteTeacherByTeaIds(Long[] teaIds);
}
```

src/main/java/com/iot/teaManager/service/ITeacherService.java

```
package com.ruoyi.teaManager.service;

import java.util.List;
import com.ruoyi.teaManager.domain.Teacher;

/**
```

```
 * 教师 Service 接口
 *
 * @author ljh
 * @date 2024-10-30
 */
public interface ITeacherService
{
    /**
     * 查询教师
     *
     * @param teald 教师主键
     * @return 教师
     */
    public Teacher selectTeacherByTeald(Long teald);

    /**
     * 查询教师列表
     *
     * @param teacher 教师
     * @return 教师集合
     */
    public List<Teacher> selectTeacherList(Teacher teacher);

    /**
     * 新增教师
     *
     * @param teacher 教师
```

```
     * @return  结果
     */
    public int insertTeacher(Teacher teacher);

    /**
     * 修改教师
     *
     * @param teacher  教师
     * @return  结果
     */
    public int updateTeacher(Teacher teacher);

    /**
     * 批量删除教师
     *
     * @param tealds  需要删除的教师主键集合
     * @return  结果
     */
    public int deleteTeacherByTealds(Long[] tealds);

    /**
     * 删除教师信息
     *
     * @param teald  教师主键
     * @return  结果
     */
    public int deleteTeacherByTeald(Long teald);
```

```
}
```

src/main/java/com/iot/teaManager/service/impl/TeacherServiceImpl.java

```
package com.ruoyi.teaManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.teaManager.mapper.TeacherMapper;
import com.ruoyi.teaManager.domain.Teacher;
import com.ruoyi.teaManager.service.ITeacherService;

/**
 * 教师 Service 业务层处理
 *
 * @author ljh
 * @date 2024-10-30
 */
@Service
public class TeacherServiceImpl implements ITeacherService
{
    @Autowired
    private TeacherMapper teacherMapper;

    /**
     * 查询教师
```

```
     *
     * @param teaId 教师主键
     * @return 教师
     */
    @Override
    public Teacher selectTeacherByTeaId(Long teaId)
    {
        return teacherMapper.selectTeacherByTeaId(teaId);
    }

    /**
     * 查询教师列表
     *
     * @param teacher 教师
     * @return 教师
     */
    @Override
    public List<Teacher> selectTeacherList(Teacher teacher)
    {
        return teacherMapper.selectTeacherList(teacher);
    }

    /**
     * 新增教师
     *
     * @param teacher 教师
     * @return 结果
```

```
     */
    @Override
    public int insertTeacher(Teacher teacher)
    {
        return teacherMapper.insertTeacher(teacher);
    }

    /**
     * 修改教师
     *
     * @param teacher 教师
     * @return 结果
     */
    @Override
    public int updateTeacher(Teacher teacher)
    {
        return teacherMapper.updateTeacher(teacher);
    }

    /**
     * 批量删除教师
     *
     * @param teaIds 需要删除的教师主键
     * @return 结果
     */
    @Override
    public int deleteTeacherByTeaIds(Long[] teaIds)
```

```
    {
        return teacherMapper.deleteTeacherByTeaIds(teaIds);
    }

    /**
     * 删除教师信息
     *
     * @param teaId 教师主键
     * @return 结果
     */
    @Override
    public int deleteTeacherByTeaId(Long teaId)
    {
        return teacherMapper.deleteTeacherByTeaId(teaId);
    }
}
```

src/main/java/com/iot/courseManager/controller/CourseController.java

```
package com.ruoyi.courseManager.controller;

import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
```

```
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.ruoyi.common.annotation.Log;
import com.ruoyi.common.core.controller.BaseController;
import com.ruoyi.common.core.domain.AjaxResult;
import com.ruoyi.common.enums.BusinessType;
import com.ruoyi.courseManager.domain.Course;
import com.ruoyi.courseManager.service.ICourseService;
import com.ruoyi.common.utils.poi.ExcelUtil;
import com.ruoyi.common.core.page.TableDataInfo;

/**
 * 课程管理 Controller
 *
 * @author LJH
 * @date 2024-11-26
 */
@RestController
@RequestMapping("/courseManager/course")
public class CourseController extends BaseController
{
    @Autowired
    private ICourseService courseService;
```

```
/**
 * 查询课程管理列表
 */
@PreAuthorize("@ss.hasPermi('courseManager:course:list')")
@GetMapping("/list")
public TableDataInfo list(Course course)
{
    startPage();
    List<Course> list = courseService.selectCourseList(course);
    return getDataTable(list);
}

/**
 * 导出课程管理列表
 */
@PreAuthorize("@ss.hasPermi('courseManager:course:export')")
@Log(title = "课程管理", businessType = BusinessType.EXPORT)
@PostMapping("/export")
public void export(HttpServletResponse response, Course course)
{
    List<Course> list = courseService.selectCourseList(course);
    ExcelUtil<Course> util = new ExcelUtil<Course>(Course.class);
    util.exportExcel(response, list, "课程管理数据");
}

/**
 * 获取课程管理详细信息
```

```
 */
@PreAuthorize("@ss.hasPermi('courseManager:course:query')")
@GetMapping(value = "/{courseId}")
public AjaxResult getInfo(@PathVariable("courseId") Long courseId)
{
        return success(courseService.selectCourseByCourseId(courseId));
}

/**
 * 新增课程管理
 */
@PreAuthorize("@ss.hasPermi('courseManager:course:add')")
@Log(title = "课程管理", businessType = BusinessType.INSERT)
@PostMapping
public AjaxResult add(@RequestBody Course course)
{
        return toAjax(courseService.insertCourse(course));
}

/**
 * 修改课程管理
 */
@PreAuthorize("@ss.hasPermi('courseManager:course:edit')")
@Log(title = "课程管理", businessType = BusinessType.UPDATE)
@PutMapping
public AjaxResult edit(@RequestBody Course course)
{
```

```
        return toAjax(courseService.updateCourse(course));
    }

    /**
     * 删除课程管理
     */
    @PreAuthorize("@ss.hasPermi('courseManager:course:remove')")
    @Log(title = "课程管理", businessType = BusinessType.DELETE)
    @DeleteMapping("/{courseIds}")
    public AjaxResult remove(@PathVariable Long[] courseIds)
    {
        return toAjax(courseService.deleteCourseByCourseIds(courseIds));
    }
}
```

src/main/java/com/iot/courseManager/domain/Course.java

```
package com.ruoyi.courseManager.domain;

import org.apache.commons.lang3.builder.ToStringBuilder;
import org.apache.commons.lang3.builder.ToStringStyle;
import com.ruoyi.common.annotation.Excel;
import com.ruoyi.common.core.domain.BaseEntity;

/**
 * 课程管理对象 course
 *
```

```
 * @author LJH
 * @date 2024-11-26
 */
public class Course extends BaseEntity
{
    private static final long serialVersionUID = 1L;

    /** 课程 id */
    private Long courseId;

    /** 教师 id */
    @Excel(name = "教师 id")
    private Long teaId;

    /** 课程号 */
    @Excel(name = "课程号")
    private String courseNum;

    /** 课程名 */
    @Excel(name = "课程名")
    private String courseName;

    public void setCourseId(Long courseId)
    {
        this.courseId = courseId;
    }
```

```
public Long getCourseId()
{
    return courseId;
}
public void setTeaId(Long teaId)
{
    this.teaId = teaId;
}

public Long getTeaId()
{
    return teaId;
}
public void setCourseNum(String courseNum)
{
    this.courseNum = courseNum;
}

public String getCourseNum()
{
    return courseNum;
}
public void setCourseName(String courseName)
{
    this.courseName = courseName;
}
```

```
    public String getCourseName()
    {
        return courseName;
    }

    @Override
    public String toString() {
        return new ToStringBuilder(this,ToStringStyle.MULTI_LINE_STYLE)
                .append("courseId", getCourseId())
                .append("teaId", getTeaId())
                .append("courseNum", getCourseNum())
                .append("courseName", getCourseName())
                .toString();
    }
}
```

src/main/java/com/iot/courseManager/mapper/CourseMapper.java

```
package com.ruoyi.courseManager.mapper;

import java.util.List;
import com.ruoyi.courseManager.domain.Course;

/**
 * 课程管理 Mapper 接口
 *
 * @author LJH
```

```
 * @date 2024-11-26
 */
public interface CourseMapper
{
    /**
     * 查询课程管理
     *
     * @param courseId 课程管理主键
     * @return 课程管理
     */
    public Course selectCourseByCourseId(Long courseId);

    /**
     * 查询课程管理列表
     *
     * @param course 课程管理
     * @return 课程管理集合
     */
    public List<Course> selectCourseList(Course course);

    /**
     * 新增课程管理
     *
     * @param course 课程管理
     * @return 结果
     */
    public int insertCourse(Course course);
```

```
    /**
     * 修改课程管理
     *
     * @param course 课程管理
     * @return 结果
     */
    public int updateCourse(Course course);

    /**
     * 删除课程管理
     *
     * @param courseId 课程管理主键
     * @return 结果
     */
    public int deleteCourseByCourseId(Long courseId);

    /**
     * 批量删除课程管理
     *
     * @param courseIds 需要删除的数据主键集合
     * @return 结果
     */
    public int deleteCourseByCourseIds(Long[] courseIds);
}
```

src/main/java/com/iot/courseManager/service/ICourseService.java

```
package com.ruoyi.courseManager.service;

import java.util.List;
import com.ruoyi.courseManager.domain.Course;

/**
 * 课程管理 Service 接口
 *
 * @author LJH
 * @date 2024-11-26
 */
public interface ICourseService
{
    /**
     * 查询课程管理
     *
     * @param courseId 课程管理主键
     * @return 课程管理
     */
    public Course selectCourseByCourseId(Long courseId);

    /**
     * 查询课程管理列表
     *
     * @param course 课程管理
     * @return 课程管理集合
```

```
 */
public List<Course> selectCourseList(Course course);

/**
 * 新增课程管理
 *
 * @param course 课程管理
 * @return 结果
 */
public int insertCourse(Course course);

/**
 * 修改课程管理
 *
 * @param course 课程管理
 * @return 结果
 */
public int updateCourse(Course course);

/**
 * 批量删除课程管理
 *
 * @param courseIds 需要删除的课程管理主键集合
 * @return 结果
 */
public int deleteCourseByCourseIds(Long[] courseIds);
```

```
    /**
     * 删除课程管理信息
     *
     * @param courseId 课程管理主键
     * @return 结果
     */
    public int deleteCourseByCourseId(Long courseId);
}
```

src/main/java/com/iot/courseManager/service/impl/CourseServiceImpl.java

```
package com.ruoyi.courseManager.service.impl;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.ruoyi.courseManager.mapper.CourseMapper;
import com.ruoyi.courseManager.domain.Course;
import com.ruoyi.courseManager.service.ICourseService;

/**
 * 课程管理 Service 业务层处理
 *
 * @author LJH
 * @date 2024-11-26
 */
@Service
```

```
public class CourseServiceImpl implements ICourseService
{
    @Autowired
    private CourseMapper courseMapper;

    /**
     * 查询课程管理
     *
     * @param courseId 课程管理主键
     * @return 课程管理
     */
    @Override
    public Course selectCourseByCourseId(Long courseId)
    {
        return courseMapper.selectCourseByCourseId(courseId);
    }

    /**
     * 查询课程管理列表
     *
     * @param course 课程管理
     * @return 课程管理
     */
    @Override
    public List<Course> selectCourseList(Course course)
    {
        return courseMapper.selectCourseList(course);
```

```
}

/**
 * 新增课程管理
 *
 * @param course 课程管理
 * @return 结果
 */
@Override
public int insertCourse(Course course)
{
    return courseMapper.insertCourse(course);
}

/**
 * 修改课程管理
 *
 * @param course 课程管理
 * @return 结果
 */
@Override
public int updateCourse(Course course)
{
    return courseMapper.updateCourse(course);
}

/**
```

```
     * 批量删除课程管理
     *
     * @param courseIds 需要删除的课程管理主键
     * @return 结果
     */
    @Override
    public int deleteCourseByCourseIds(Long[] courseIds)
    {
        return courseMapper.deleteCourseByCourseIds(courseIds);
    }

    /**
     * 删除课程管理信息
     *
     * @param courseId 课程管理主键
     * @return 结果
     */
    @Override
    public int deleteCourseByCourseId(Long courseId)
    {
        return courseMapper.deleteCourseByCourseId(courseId);
    }
}
```

src/view/courseManager/course/index.vue

```
<template>
  <div class="app-container">
```

```
<el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-show="showSearch" label-width="68px">
    <el-form-item label="教师 id" prop="teaId">
      <el-input
        v-model="queryParams.teaId"
        placeholder="请输入教师 id"
        clearable
        @keyup.enter.native="handleQuery"
      />
    </el-form-item>
    <el-form-item label="课程号" prop="courseNum">
      <el-input
        v-model="queryParams.courseNum"
        placeholder="请输入课程号"
        clearable
        @keyup.enter.native="handleQuery"
      />
    </el-form-item>
    <el-form-item label="课程名" prop="courseName">
      <el-input
        v-model="queryParams.courseName"
        placeholder="请输入课程名"
        clearable
        @keyup.enter.native="handleQuery"
      />
    </el-form-item>
    <el-form-item>
```

```
            <el-button type="primary" icon="el-icon-search" size="mini"
@click="handleQuery">搜索</el-button>
            <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置
</el-button>
          </el-form-item>
        </el-form>

        <el-row :gutter="10" class="mb8">
          <el-col :span="1.5">
            <el-button
              type="primary"
              plain
              icon="el-icon-plus"
              size="mini"
              @click="handleAdd"
              v-hasPermi="['courseManager:course:add']"
            >新增</el-button>
          </el-col>
          <el-col :span="1.5">
            <el-button
              type="success"
              plain
              icon="el-icon-edit"
              size="mini"
              :disabled="single"
              @click="handleUpdate"
              v-hasPermi="['courseManager:course:edit']"
```

```
        >修改</el-button>
      </el-col>
      <el-col :span="1.5">
        <el-button
          type="danger"
          plain
          icon="el-icon-delete"
          size="mini"
          :disabled="multiple"
          @click="handleDelete"
          v-hasPermi="['courseManager:course:remove']"
        >删除</el-button>
      </el-col>
      <el-col :span="1.5">
        <el-button
          type="warning"
          plain
          icon="el-icon-download"
          size="mini"
          @click="handleExport"
          v-hasPermi="['courseManager:course:export']"
        >导出</el-button>
      </el-col>
      <right-toolbar :showSearch.sync="showSearch"
@queryTable="getList"></right-toolbar>
    </el-row>
```

```
<el-table v-loading="loading" :data="courseList" @selection-change="handleSelectionChange">
    <el-table-column type="selection" width="55" align="center" />
    <el-table-column label="课程 id" align="center" prop="courseId" />
    <el-table-column label="教师 id" align="center" prop="teaId" />
    <el-table-column label="课程号" align="center" prop="courseNum" />
    <el-table-column label="课程名" align="center" prop="courseName" />
    <el-table-column label="操作" align="center" class-name="small-padding fixed-width">
      <template slot-scope="scope">
        <el-button
          size="mini"
          type="text"
          icon="el-icon-edit"
          @click="handleUpdate(scope.row)"
          v-hasPermi="['courseManager:course:edit']"
        >修改</el-button>
        <el-button
          size="mini"
          type="text"
          icon="el-icon-delete"
          @click="handleDelete(scope.row)"
          v-hasPermi="['courseManager:course:remove']"
        >删除</el-button>
      </template>
    </el-table-column>
  </el-table>
```

```
<pagination
    v-show="total>0"
    :total="total"
    :page.sync="queryParams.pageNum"
    :limit.sync="queryParams.pageSize"
    @pagination="getList"
/>

<!-- 添加或修改课程管理对话框 -->
<el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
    <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="教师 id" prop="teaId">
            <el-input v-model="form.teaId" placeholder="请输入教师 id" />
        </el-form-item>
        <el-form-item label="课程号" prop="courseNum">
            <el-input v-model="form.courseNum" placeholder="请输入课程号" />
        </el-form-item>
        <el-form-item label="课程名" prop="courseName">
            <el-input v-model="form.courseName" placeholder="请输入课程名" />
        </el-form-item>
    </el-form>
    <div slot="footer" class="dialog-footer">
        <el-button type="primary" @click="submitForm">确 定</el-button>
        <el-button @click="cancel">取 消</el-button>
    </div>
</el-dialog>
```

```
    </div>
</template>

<script>
import { listCourse, getCourse, delCourse, addCourse, updateCourse } from
"@/api/courseManager/course";

export default {
    name: "Course",
    data() {
        return {
            // 遮罩层
            loading: true,
            // 选中数组
            ids: [],
            // 非单个禁用
            single: true,
            // 非多个禁用
            multiple: true,
            // 显示搜索条件
            showSearch: true,
            // 总条数
            total: 0,
            // 课程管理表格数据
            courseList: [],
            // 弹出层标题
            title: "",
```

```
        // 是否显示弹出层
        open: false,
        // 查询参数
        queryParams: {
            pageNum: 1,
            pageSize: 10,
            teaId: null,
            courseNum: null,
            courseName: null
        },
        // 表单参数
        form: {},
        // 表单校验
        rules: {
        }
    };
},
created() {
    this.getList();
},
methods: {
    /** 查询课程管理列表 */
    getList() {
        this.loading = true;
        listCourse(this.queryParams).then(response => {
            this.courseList = response.rows;
            this.total = response.total;
```

```
        this.loading = false;
      });
    },
    // 取消按钮
    cancel() {
      this.open = false;
      this.reset();
    },
    // 表单重置
    reset() {
      this.form = {
        courseId: null,
        teaId: null,
        courseNum: null,
        courseName: null
      };
      this.resetForm("form");
    },
    /** 搜索按钮操作 */
    handleQuery() {
      this.queryParams.pageNum = 1;
      this.getList();
    },
    /** 重置按钮操作 */
    resetQuery() {
      this.resetForm("queryForm");
      this.handleQuery();
```

```
},
// 多选框选中数据
handleSelectionChange(selection) {
    this.ids = selection.map(item => item.courseId)
    this.single = selection.length!==1
    this.multiple = !selection.length
},
/** 新增按钮操作 */
handleAdd() {
    this.reset();
    this.open = true;
    this.title = "添加课程管理";
},
/** 修改按钮操作 */
handleUpdate(row) {
    this.reset();
    const courseId = row.courseId || this.ids
    getCourse(courseId).then(response => {
        this.form = response.data;
        this.open = true;
        this.title = "修改课程管理";
    });
},
/** 提交按钮 */
submitForm() {
    this.$refs["form"].validate(valid => {
        if (valid) {
```

```
                if (this.form.courseId != null) {
                    updateCourse(this.form).then(response => {
                        this.$modal.msgSuccess("修改成功");
                        this.open = false;
                        this.getList();
                    });
                } else {
                    addCourse(this.form).then(response => {
                        this.$modal.msgSuccess("新增成功");
                        this.open = false;
                        this.getList();
                    });
                }
            }
        });
    },
    /** 删除按钮操作 */
    handleDelete(row) {
        const courseIds = row.courseId || this.ids;
        this.$modal.confirm('是否确认删除课程管理编号为"' + courseIds + '"的数据项？').then(function() {
            return delCourse(courseIds);
        }).then(() => {
            this.getList();
            this.$modal.msgSuccess("删除成功");
        }).catch(() => {});
    },
```

```
    /** 导出按钮操作 */
    handleExport() {
      this.download('courseManager/course/export', {
        ...this.queryParams
      }, `course_${new Date().getTime()}.xlsx`)
    }
  }
};
</script>
```

src/view/machineManager/machine/index.vue

```
<template>
  <div class="app-container">
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-show="showSearch" label-width="68px">
      <el-form-item label="机器编号" prop="macNum">
        <el-input
          v-model="queryParams.macNum"
          placeholder="请输入机器编号"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="机器名称" prop="macName">
        <el-input
          v-model="queryParams.macName"
          placeholder="请输入机器名称"
```

```
            clearable
            @keyup.enter.native="handleQuery"
          />
        </el-form-item>
        <el-form-item label="机器类型" prop="macType">
          <el-select v-model="queryParams.macType" placeholder="请选择机器类型" clearable>
            <el-option
              v-for="dict in dict.type.mac_type"
              :key="dict.value"
              :label="dict.label"
              :value="dict.value"
            />
          </el-select>
        </el-form-item>
        <el-form-item>
          <el-button type="primary" icon="el-icon-search" size="mini" @click="handleQuery">搜索</el-button>
          <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置</el-button>
        </el-form-item>
      </el-form>

      <el-row :gutter="10" class="mb8">
        <el-col :span="1.5">
          <el-button
            type="primary"
            plain
```

```
        icon="el-icon-plus"
        size="mini"
        @click="handleAdd"
        v-hasPermi="['machineManager:machine:add']"
      >新增</el-button>
    </el-col>
    <el-col :span="1.5">
      <el-button
        type="success"
        plain
        icon="el-icon-edit"
        size="mini"
        :disabled="single"
        @click="handleUpdate"
        v-hasPermi="['machineManager:machine:edit']"
      >修改</el-button>
    </el-col>
    <el-col :span="1.5">
      <el-button
        type="danger"
        plain
        icon="el-icon-delete"
        size="mini"
        :disabled="multiple"
        @click="handleDelete"
        v-hasPermi="['machineManager:machine:remove']"
      >删除</el-button>
```

```
      </el-col>
      <el-col :span="1.5">
        <el-button
          type="warning"
          plain
          icon="el-icon-download"
          size="mini"
          @click="handleExport"
          v-hasPermi="['machineManager:machine:export']"
        >导出</el-button>
      </el-col>
      <right-toolbar :showSearch.sync="showSearch" @queryTable="getList"></right-toolbar>
    </el-row>

    <el-table v-loading="loading" :data="machineList" @selection-change="handleSelectionChange">
      <el-table-column type="selection" width="55" align="center" />
      <el-table-column label="机器 id" align="center" prop="macId" />
      <el-table-column label="机器编号" align="center" prop="macNum" />
      <el-table-column label="机器名称" align="center" prop="macName" />
      <el-table-column label="机器类型" align="center" prop="macType">
        <template slot-scope="scope">
          <dict-tag :options="dict.type.mac_type" :value="scope.row.macType"/>
        </template>
      </el-table-column>
      <el-table-column label="操作" align="center" class-name="small-padding fixed-width">
```

```
        <template slot-scope="scope">
          <el-button
            size="mini"
            type="text"
            icon="el-icon-edit"
            @click="handleUpdate(scope.row)"
            v-hasPermi="['machineManager:machine:edit']"
          >修改</el-button>
          <el-button
            size="mini"
            type="text"
            icon="el-icon-delete"
            @click="handleDelete(scope.row)"
            v-hasPermi="['machineManager:machine:remove']"
          >删除</el-button>
        </template>
      </el-table-column>
    </el-table>

    <pagination
      v-show="total>0"
      :total="total"
      :page.sync="queryParams.pageNum"
      :limit.sync="queryParams.pageSize"
      @pagination="getList"
    />
```

```
    <!-- 添加或修改机器管理对话框 -->
    <el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
      <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="机器编号" prop="macNum">
          <el-input v-model="form.macNum" placeholder="请输入机器编号" />
        </el-form-item>
        <el-form-item label="机器名称" prop="macName">
          <el-input v-model="form.macName" placeholder="请输入机器名称" />
        </el-form-item>
        <el-form-item label="机器类型" prop="macType">
          <el-select v-model="form.macType" placeholder="请选择机器类型">
            <el-option
              v-for="dict in dict.type.mac_type"
              :key="dict.value"
              :label="dict.label"
              :value="dict.value"
            ></el-option>
          </el-select>
        </el-form-item>
      </el-form>
      <div slot="footer" class="dialog-footer">
        <el-button type="primary" @click="submitForm">确 定</el-button>
        <el-button @click="cancel">取 消</el-button>
      </div>
    </el-dialog>
  </div>
</template>
```

```
<script>
import { listMachine, getMachine, delMachine, addMachine, updateMachine } from "@/api/machineManager/machine";

export default {
  name: "Machine",
  dicts: ['mac_type'],
  data() {
    return {
      // 遮罩层
      loading: true,
      // 选中数组
      ids: [],
      // 非单个禁用
      single: true,
      // 非多个禁用
      multiple: true,
      // 显示搜索条件
      showSearch: true,
      // 总条数
      total: 0,
      // 机器管理表格数据
      machineList: [],
      // 弹出层标题
      title: "",
      // 是否显示弹出层
```

```
        open: false,
        // 查询参数
        queryParams: {
          pageNum: 1,
          pageSize: 10,
          macNum: null,
          macName: null,
          macType: null
        },
        // 表单参数
        form: {},
        // 表单校验
        rules: {
          macNum: [
            { required: true, message: "机器编号不能为空", trigger: "blur" }
          ],
          macName: [
            { required: true, message: "机器名称不能为空", trigger: "blur" }
          ],
          macType: [
            { required: true, message: "机器类型不能为空", trigger: "change" }
          ]
        }
      };
    },
    created() {
      this.getList();
```

```
},
methods: {
  /** 查询机器管理列表 */
  getList() {
    this.loading = true;
    listMachine(this.queryParams).then(response => {
      this.machineList = response.rows;
      this.total = response.total;
      this.loading = false;
    });
  },
  // 取消按钮
  cancel() {
    this.open = false;
    this.reset();
  },
  // 表单重置
  reset() {
    this.form = {
      macId: null,
      macNum: null,
      macName: null,
      macType: null
    };
    this.resetForm("form");
  },
  /** 搜索按钮操作 */
```

```
handleQuery() {
    this.queryParams.pageNum = 1;
    this.getList();
},
/** 重置按钮操作 */
resetQuery() {
    this.resetForm("queryForm");
    this.handleQuery();
},
// 多选框选中数据
handleSelectionChange(selection) {
    this.ids = selection.map(item => item.macId)
    this.single = selection.length!==1
    this.multiple = !selection.length
},
/** 新增按钮操作 */
handleAdd() {
    this.reset();
    this.open = true;
    this.title = "添加机器管理";
},
/** 修改按钮操作 */
handleUpdate(row) {
    this.reset();
    const macId = row.macId || this.ids
    getMachine(macId).then(response => {
        this.form = response.data;
```

```
        this.open = true;
        this.title = "修改机器管理";
    });
},
/** 提交按钮 */
submitForm() {
    this.$refs["form"].validate(valid => {
        if (valid) {
            if (this.form.macId != null) {
                updateMachine(this.form).then(response => {
                    this.$modal.msgSuccess("修改成功");
                    this.open = false;
                    this.getList();
                });
            } else {
                addMachine(this.form).then(response => {
                    this.$modal.msgSuccess("新增成功");
                    this.open = false;
                    this.getList();
                });
            }
        }
    });
},
/** 删除按钮操作 */
handleDelete(row) {
    const macIds = row.macId || this.ids;
```

```
        this.$modal.confirm('是否确认删除机器管理编号为"' + macIds + '"的数据项?').then(function() {
            return delMachine(macIds);
        }).then(() => {
            this.getList();
            this.$modal.msgSuccess("删除成功");
        }).catch(() => {});
    },
    /** 导出按钮操作 */
    handleExport() {
        this.download('machineManager/machine/export', {
            ...this.queryParams
        }, `machine_${new Date().getTime()}.xlsx`)
    }
  }
};
</script>
```

src/view/phytestManager/phytest/index.vue

```
<template>
  <div class="app-container">
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-show="showSearch" label-width="68px">
      <el-form-item label="项目序号" prop="ptNum">
        <el-input
          v-model="queryParams.ptNum"
          placeholder="请输入项目序号"
```

```
            clearable
            @keyup.enter.native="handleQuery"
          />
        </el-form-item>
        <el-form-item label="项目名称" prop="ptName">
          <el-input
            v-model="queryParams.ptName"
            placeholder="请输入项目名称"
            clearable
            @keyup.enter.native="handleQuery"
          />
        </el-form-item>
        <el-form-item label="项目权重" prop="ptWeight">
          <el-input
            v-model="queryParams.ptWeight"
            placeholder="请输入项目权重"
            clearable
            @keyup.enter.native="handleQuery"
          />
        </el-form-item>
        <el-form-item>
          <el-button type="primary" icon="el-icon-search" size="mini"
@click="handleQuery">搜索</el-button>
          <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置
</el-button>
        </el-form-item>
      </el-form>
```

```
<el-row :gutter="10" class="mb8">
  <el-col :span="1.5">
    <el-button
      type="primary"
      plain
      icon="el-icon-plus"
      size="mini"
      @click="handleAdd"
      v-hasPermi="['phytestManager:phytest:add']"
    >新增</el-button>
  </el-col>
  <el-col :span="1.5">
    <el-button
      type="success"
      plain
      icon="el-icon-edit"
      size="mini"
      :disabled="single"
      @click="handleUpdate"
      v-hasPermi="['phytestManager:phytest:edit']"
    >修改</el-button>
  </el-col>
  <el-col :span="1.5">
    <el-button
      type="danger"
      plain
```

```
            icon="el-icon-delete"
            size="mini"
            :disabled="multiple"
            @click="handleDelete"
            v-hasPermi="['phytestManager:phytest:remove']"
          >删除</el-button>
        </el-col>
        <el-col :span="1.5">
          <el-button
            type="warning"
            plain
            icon="el-icon-download"
            size="mini"
            @click="handleExport"
            v-hasPermi="['phytestManager:phytest:export']"
          >导出</el-button>
        </el-col>
        <right-toolbar :showSearch.sync="showSearch" @queryTable="getList"></right-toolbar>
      </el-row>

      <el-table v-loading="loading" :data="phytestList" @selection-change="handleSelectionChange">
        <el-table-column type="selection" width="55" align="center" />
        <el-table-column label="项目 id" align="center" prop="ptId" />
        <el-table-column label="项目序号" align="center" prop="ptNum" />
        <el-table-column label="项目名称" align="center" prop="ptName" />
```

```
        <el-table-column label="项目权重" align="center" prop="ptWeight" />
        <el-table-column label="操作" align="center" class-name="small-padding fixed-width">
          <template slot-scope="scope">
            <el-button
              size="mini"
              type="text"
              icon="el-icon-edit"
              @click="handleUpdate(scope.row)"
              v-hasPermi="['phytestManager:phytest:edit']"
            >修改</el-button>
            <el-button
              size="mini"
              type="text"
              icon="el-icon-delete"
              @click="handleDelete(scope.row)"
              v-hasPermi="['phytestManager:phytest:remove']"
            >删除</el-button>
          </template>
        </el-table-column>
      </el-table>

      <pagination
        v-show="total>0"
        :total="total"
        :page.sync="queryParams.pageNum"
        :limit.sync="queryParams.pageSize"
```

```
      @pagination="getList"
    />

    <!-- 添加或修改体测项目管理对话框 -->
    <el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
      <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="项目序号" prop="ptNum">
          <el-input v-model="form.ptNum" placeholder="请输入项目序号" />
        </el-form-item>
        <el-form-item label="项目名称" prop="ptName">
          <el-input v-model="form.ptName" placeholder="请输入项目名称" />
        </el-form-item>
        <el-form-item label="项目权重" prop="ptWeight">
          <el-input v-model="form.ptWeight" placeholder="请输入项目权重" />
        </el-form-item>
      </el-form>
      <div slot="footer" class="dialog-footer">
        <el-button type="primary" @click="submitForm">确 定</el-button>
        <el-button @click="cancel">取 消</el-button>
      </div>
    </el-dialog>
  </div>
</template>

<script>
import { listPhytest, getPhytest, delPhytest, addPhytest, updatePhytest } from
"@/api/phytestManager/phytest";
```

```
export default {
    name: "Phytest",
    data() {
        return {
            // 遮罩层
            loading: true,
            // 选中数组
            ids: [],
            // 非单个禁用
            single: true,
            // 非多个禁用
            multiple: true,
            // 显示搜索条件
            showSearch: true,
            // 总条数
            total: 0,
            // 体测项目管理表格数据
            phytestList: [],
            // 弹出层标题
            title: "",
            // 是否显示弹出层
            open: false,
            // 查询参数
            queryParams: {
                pageNum: 1,
                pageSize: 10,
```

```
        ptNum: null,
        ptName: null,
        ptWeight: null
      },
      // 表单参数
      form: {},
      // 表单校验
      rules: {
      }
    };
  },
  created() {
    this.getList();
  },
  methods: {
    /** 查询体测项目管理列表 */
    getList() {
      this.loading = true;
      listPhytest(this.queryParams).then(response => {
        this.phytestList = response.rows;
        this.total = response.total;
        this.loading = false;
      });
    },
    // 取消按钮
    cancel() {
      this.open = false;
```

```
    this.reset();
},
// 表单重置
reset() {
    this.form = {
        ptId: null,
        ptNum: null,
        ptName: null,
        ptWeight: null
    };
    this.resetForm("form");
},
/** 搜索按钮操作 */
handleQuery() {
    this.queryParams.pageNum = 1;
    this.getList();
},
/** 重置按钮操作 */
resetQuery() {
    this.resetForm("queryForm");
    this.handleQuery();
},
// 多选框选中数据
handleSelectionChange(selection) {
    this.ids = selection.map(item => item.ptId)
    this.single = selection.length!==1
    this.multiple = !selection.length
```

```
},
/** 新增按钮操作 */
handleAdd() {
  this.reset();
  this.open = true;
  this.title = "添加体测项目管理";
},
/** 修改按钮操作 */
handleUpdate(row) {
  this.reset();
  const ptId = row.ptId || this.ids
  getPhytest(ptId).then(response => {
    this.form = response.data;
    this.open = true;
    this.title = "修改体测项目管理";
  });
},
/** 提交按钮 */
submitForm() {
  this.$refs["form"].validate(valid => {
    if (valid) {
      if (this.form.ptId != null) {
        updatePhytest(this.form).then(response => {
          this.$modal.msgSuccess("修改成功");
          this.open = false;
          this.getList();
        });
```

```
            } else {
                addPhytest(this.form).then(response => {
                    this.$modal.msgSuccess("新增成功");
                    this.open = false;
                    this.getList();
                });
            }
        }
      });
    },
    /** 删除按钮操作 */
    handleDelete(row) {
      const ptIds = row.ptId || this.ids;
      this.$modal.confirm('是否确认删除体测项目管理编号为"' + ptIds + '"的数据项?').then(function() {
        return delPhytest(ptIds);
      }).then(() => {
        this.getList();
        this.$modal.msgSuccess("删除成功");
      }).catch(() => {});
    },
    /** 导出按钮操作 */
    handleExport() {
      this.download('phytestManager/phytest/export', {
        ...this.queryParams
      }, `phytest_${new Date().getTime()}.xlsx`)
    }
```

```
  }
};
</script>
```

src/view/ptbookManager/ptbook/index.vue

```
<template>
  <div class="app-container">
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-show="showSearch" label-width="68px">
      <el-form-item label="容量" prop="ptbookCon">
        <el-input
          v-model="queryParams.ptbookCon"
          placeholder="请输入容量"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="已约人数" prop="ptbookApnum">
        <el-input
          v-model="queryParams.ptbookApnum"
          placeholder="请输入已约人数"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="预约开始日期" prop="ptbookBegintime">
        <el-date-picker clearable
```

```
            v-model="queryParams.ptbookBegintime"
            type="date"
            value-format="yyyy-MM-dd"
            placeholder="请选择预约开始日期">
          </el-date-picker>
        </el-form-item>
        <el-form-item label="测试时间" prop="ptbookTesttime">
          <el-date-picker clearable
            v-model="queryParams.ptbookTesttime"
            type="date"
            value-format="yyyy-MM-dd"
            placeholder="请选择测试时间">
          </el-date-picker>
        </el-form-item>
        <el-form-item label="预约结束日期" prop="ptbookEndtime">
          <el-date-picker clearable
            v-model="queryParams.ptbookEndtime"
            type="date"
            value-format="yyyy-MM-dd"
            placeholder="请选择预约结束日期">
          </el-date-picker>
        </el-form-item>
        <el-form-item>
          <el-button type="primary" icon="el-icon-search" size="mini"
@click="handleQuery">搜索</el-button>
          <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置
</el-button>
```

```
    </el-form-item>
</el-form>

<el-row :gutter="10" class="mb8">
    <el-col :span="1.5">
        <el-button
            type="primary"
            plain
            icon="el-icon-plus"
            size="mini"
            @click="handleAdd"
            v-hasPermi="['ptbookManager:ptbook:add']"
        >新增</el-button>
    </el-col>
    <el-col :span="1.5">
        <el-button
            type="success"
            plain
            icon="el-icon-edit"
            size="mini"
            :disabled="single"
            @click="handleUpdate"
            v-hasPermi="['ptbookManager:ptbook:edit']"
        >修改</el-button>
    </el-col>
    <el-col :span="1.5">
        <el-button
```

```
            type="danger"
            plain
            icon="el-icon-delete"
            size="mini"
            :disabled="multiple"
            @click="handleDelete"
            v-hasPermi="['ptbookManager:ptbook:remove']"
          >删除</el-button>
        </el-col>
        <el-col :span="1.5">
          <el-button
            type="warning"
            plain
            icon="el-icon-download"
            size="mini"
            @click="handleExport"
            v-hasPermi="['ptbookManager:ptbook:export']"
          >导出</el-button>
        </el-col>
        <right-toolbar :showSearch.sync="showSearch" @queryTable="getList"></right-toolbar>
      </el-row>

      <el-table v-loading="loading" :data="ptbookList" @selection-change="handleSelectionChange">
        <el-table-column type="selection" width="55" align="center" />
        <el-table-column label="预约 id" align="center" prop="ptbookId" />
```

```
<el-table-column label="容量" align="center" prop="ptbookCon" />
<el-table-column label="已约人数" align="center" prop="ptbookApnum" />
<el-table-column label="预约开始日期" align="center" prop="ptbookBegintime" width="180">
  <template slot-scope="scope">
    <span>{{ parseTime(scope.row.ptbookBegintime, '{y}-{m}-{d}') }}</span>
  </template>
</el-table-column>
<el-table-column label="测试时间" align="center" prop="ptbookTesttime" width="180">
  <template slot-scope="scope">
    <span>{{ parseTime(scope.row.ptbookTesttime, '{y}-{m}-{d}') }}</span>
  </template>
</el-table-column>
<el-table-column label="预约结束日期" align="center" prop="ptbookEndtime" width="180">
  <template slot-scope="scope">
    <span>{{ parseTime(scope.row.ptbookEndtime, '{y}-{m}-{d}') }}</span>
  </template>
</el-table-column>
<el-table-column label="操作" align="center" class-name="small-padding fixed-width">
  <template slot-scope="scope">
    <el-button
      size="mini"
      type="text"
      icon="el-icon-edit"
      @click="handleUpdate(scope.row)"
```

```
            v-hasPermi="['ptbookManager:ptbook:edit']"
          >修改</el-button>
          <el-button
            size="mini"
            type="text"
            icon="el-icon-delete"
            @click="handleDelete(scope.row)"
            v-hasPermi="['ptbookManager:ptbook:remove']"
          >删除</el-button>
        </template>
      </el-table-column>
    </el-table>

    <pagination
      v-show="total>0"
      :total="total"
      :page.sync="queryParams.pageNum"
      :limit.sync="queryParams.pageSize"
      @pagination="getList"
    />

    <!-- 添加或修改体测预约管理对话框 -->
    <el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
      <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="容量" prop="ptbookCon">
          <el-input v-model="form.ptbookCon" placeholder="请输入容量" />
        </el-form-item>
```

```
            <el-form-item label="已约人数" prop="ptbookApnum">
              <el-input v-model="form.ptbookApnum" placeholder="请输入已约人数"
/>
            </el-form-item>
            <el-form-item label="预约开始日期" prop="ptbookBegintime">
              <el-date-picker clearable
                v-model="form.ptbookBegintime"
                type="date"
                value-format="yyyy-MM-dd"
                placeholder="请选择预约开始日期">
              </el-date-picker>
            </el-form-item>
            <el-form-item label="测试时间" prop="ptbookTesttime">
              <el-date-picker clearable
                v-model="form.ptbookTesttime"
                type="date"
                value-format="yyyy-MM-dd"
                placeholder="请选择测试时间">
              </el-date-picker>
            </el-form-item>
            <el-form-item label="预约结束日期" prop="ptbookEndtime">
              <el-date-picker clearable
                v-model="form.ptbookEndtime"
                type="date"
                value-format="yyyy-MM-dd"
                placeholder="请选择预约结束日期">
              </el-date-picker>
```

```
            </el-form-item>
          </el-form>
          <div slot="footer" class="dialog-footer">
            <el-button type="primary" @click="submitForm">确 定</el-button>
            <el-button @click="cancel">取 消</el-button>
          </div>
        </el-dialog>
    </div>
</template>

<script>
import { listPtbook, getPtbook, delPtbook, addPtbook, updatePtbook } from "@/api/ptbookManager/ptbook";

export default {
    name: "Ptbook",
    data() {
        return {
            // 遮罩层
            loading: true,
            // 选中数组
            ids: [],
            // 非单个禁用
            single: true,
            // 非多个禁用
            multiple: true,
            // 显示搜索条件
```

```
        showSearch: true,
        // 总条数
        total: 0,
        // 体测预约管理表格数据
        ptbookList: [],
        // 弹出层标题
        title: "",
        // 是否显示弹出层
        open: false,
        // 查询参数
        queryParams: {
          pageNum: 1,
          pageSize: 10,
          ptbookCon: null,
          ptbookApnum: null,
          ptbookBegintime: null,
          ptbookTesttime: null,
          ptbookEndtime: null
        },
        // 表单参数
        form: {},
        // 表单校验
        rules: {
        }
      };
  },
  created() {
```

```
    this.getList();
},
methods: {
    /** 查询体测预约管理列表 */
    getList() {
        this.loading = true;
        listPtbook(this.queryParams).then(response => {
            this.ptbookList = response.rows;
            this.total = response.total;
            this.loading = false;
        });
    },
    // 取消按钮
    cancel() {
        this.open = false;
        this.reset();
    },
    // 表单重置
    reset() {
        this.form = {
            ptbookId: null,
            ptbookCon: null,
            ptbookApnum: null,
            ptbookBegintime: null,
            ptbookTesttime: null,
            ptbookEndtime: null
        };
```

```
    this.resetForm("form");
  },
  /** 搜索按钮操作 */
  handleQuery() {
    this.queryParams.pageNum = 1;
    this.getList();
  },
  /** 重置按钮操作 */
  resetQuery() {
    this.resetForm("queryForm");
    this.handleQuery();
  },
  // 多选框选中数据
  handleSelectionChange(selection) {
    this.ids = selection.map(item => item.ptbookId)
    this.single = selection.length!==1
    this.multiple = !selection.length
  },
  /** 新增按钮操作 */
  handleAdd() {
    this.reset();
    this.open = true;
    this.title = "添加体测预约管理";
  },
  /** 修改按钮操作 */
  handleUpdate(row) {
    this.reset();
```

```
    const ptbookId = row.ptbookId || this.ids
    getPtbook(ptbookId).then(response => {
      this.form = response.data;
      this.open = true;
      this.title = "修改体测预约管理";
    });
  },
  /** 提交按钮 */
  submitForm() {
    this.$refs["form"].validate(valid => {
      if (valid) {
        if (this.form.ptbookId != null) {
          updatePtbook(this.form).then(response => {
            this.$modal.msgSuccess("修改成功");
            this.open = false;
            this.getList();
          });
        } else {
          addPtbook(this.form).then(response => {
            this.$modal.msgSuccess("新增成功");
            this.open = false;
            this.getList();
          });
        }
      }
    });
  },
```

```
    /** 删除按钮操作 */
    handleDelete(row) {
      const ptbookIds = row.ptbookId || this.ids;
      this.$modal.confirm('是否确认删除体测预约管理编号为"' + ptbookIds + '"的数据项？').then(function() {
        return delPtbook(ptbookIds);
      }).then(() => {
        this.getList();
        this.$modal.msgSuccess("删除成功");
      }).catch(() => {});
    },
    /** 导出按钮操作 */
    handleExport() {
      this.download('ptbookManager/ptbook/export', {
        ...this.queryParams
      }, `ptbook_${new Date().getTime()}.xlsx`)
    }
  }
};
</script>
```

src/view/ptrecordManager/ptrecord/index.vue

```
<template>
  <div class="app-container">
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-show="showSearch" label-width="68px">
      <el-form-item label="机器 id" prop="macId">
```

```
    <el-input
      v-model="queryParams.macId"
      placeholder="请输入机器 id"
      clearable
      @keyup.enter.native="handleQuery"
    />
  </el-form-item>
  <el-form-item label="管理员 id" prop="adminId">
    <el-input
      v-model="queryParams.adminId"
      placeholder="请输入管理员 id"
      clearable
      @keyup.enter.native="handleQuery"
    />
  </el-form-item>
  <el-form-item label="项目 id" prop="ptId">
    <el-input
      v-model="queryParams.ptId"
      placeholder="请输入项目 id"
      clearable
      @keyup.enter.native="handleQuery"
    />
  </el-form-item>
  <el-form-item label="学生 id" prop="stuId">
    <el-input
      v-model="queryParams.stuId"
      placeholder="请输入学生 id"
```

```
      clearable
      @keyup.enter.native="handleQuery"
    />
  </el-form-item>
  <el-form-item label="记录序号" prop="ptrNum">
    <el-input
      v-model="queryParams.ptrNum"
      placeholder="请输入记录序号"
      clearable
      @keyup.enter.native="handleQuery"
    />
  </el-form-item>
  <el-form-item label="成绩" prop="ptrScore">
    <el-input
      v-model="queryParams.ptrScore"
      placeholder="请输入成绩"
      clearable
      @keyup.enter.native="handleQuery"
    />
  </el-form-item>
  <el-form-item label="权重分" prop="ptrWscore">
    <el-input
      v-model="queryParams.ptrWscore"
      placeholder="请输入权重分"
      clearable
      @keyup.enter.native="handleQuery"
    />
```

```
      </el-form-item>
      <el-form-item label="测试日期" prop="ptrDate">
        <el-date-picker clearable
          v-model="queryParams.ptrDate"
          type="date"
          value-format="yyyy-MM-dd"
          placeholder="请选择测试日期">
        </el-date-picker>
      </el-form-item>
      <el-form-item>
        <el-button type="primary" icon="el-icon-search" size="mini" @click="handleQuery">搜索</el-button>
        <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置</el-button>
      </el-form-item>
    </el-form>

    <el-row :gutter="10" class="mb8">
      <el-col :span="1.5">
        <el-button
          type="primary"
          plain
          icon="el-icon-plus"
          size="mini"
          @click="handleAdd"
          v-hasPermi="['ptrecordManager:ptrecord:add']"
        >新增</el-button>
```

```
</el-col>
<el-col :span="1.5">
  <el-button
    type="success"
    plain
    icon="el-icon-edit"
    size="mini"
    :disabled="single"
    @click="handleUpdate"
    v-hasPermi="['ptrecordManager:ptrecord:edit']"
  >修改</el-button>
</el-col>
<el-col :span="1.5">
  <el-button
    type="danger"
    plain
    icon="el-icon-delete"
    size="mini"
    :disabled="multiple"
    @click="handleDelete"
    v-hasPermi="['ptrecordManager:ptrecord:remove']"
  >删除</el-button>
</el-col>
<el-col :span="1.5">
  <el-button
    type="warning"
    plain
```

```
            icon="el-icon-download"
            size="mini"
            @click="handleExport"
            v-hasPermi="['ptrecordManager:ptrecord:export']"
          >导出</el-button>
        </el-col>
        <right-toolbar :showSearch.sync="showSearch" @queryTable="getList"></right-toolbar>
      </el-row>

      <el-table v-loading="loading" :data="ptrecordList" @selection-change="handleSelectionChange">
        <el-table-column type="selection" width="55" align="center" />
        <el-table-column label="记录 id" align="center" prop="ptrId" />
        <el-table-column label="机器 id" align="center" prop="macId" />
        <el-table-column label="管理员 id" align="center" prop="adminId" />
        <el-table-column label="项目 id" align="center" prop="ptId" />
        <el-table-column label="学生 id" align="center" prop="stuId" />
        <el-table-column label="记录序号" align="center" prop="ptrNum" />
        <el-table-column label="成绩" align="center" prop="ptrScore" />
        <el-table-column label="权重分" align="center" prop="ptrWscore" />
        <el-table-column label="测试日期" align="center" prop="ptrDate" width="180">
          <template slot-scope="scope">
            <span>{{ parseTime(scope.row.ptrDate, '{y}-{m}-{d}') }}</span>
          </template>
        </el-table-column>
        <el-table-column label="操作" align="center" class-name="small-padding
```

```
fixed-width">
            <template slot-scope="scope">
              <el-button
                size="mini"
                type="text"
                icon="el-icon-edit"
                @click="handleUpdate(scope.row)"
                v-hasPermi="['ptrecordManager:ptrecord:edit']"
              >修改</el-button>
              <el-button
                size="mini"
                type="text"
                icon="el-icon-delete"
                @click="handleDelete(scope.row)"
                v-hasPermi="['ptrecordManager:ptrecord:remove']"
              >删除</el-button>
            </template>
          </el-table-column>
        </el-table>

        <pagination
          v-show="total>0"
          :total="total"
          :page.sync="queryParams.pageNum"
          :limit.sync="queryParams.pageSize"
          @pagination="getList"
        />
```

```
<!-- 添加或修改体测记录管理对话框 -->
<el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
    <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="机器 id" prop="macId">
            <el-input v-model="form.macId" placeholder="请输入机器 id" />
        </el-form-item>
        <el-form-item label="管理员 id" prop="adminId">
            <el-input v-model="form.adminId" placeholder="请输入管理员 id" />
        </el-form-item>
        <el-form-item label="项目 id" prop="ptId">
            <el-input v-model="form.ptId" placeholder="请输入项目 id" />
        </el-form-item>
        <el-form-item label="学生 id" prop="stuId">
            <el-input v-model="form.stuId" placeholder="请输入学生 id" />
        </el-form-item>
        <el-form-item label="记录序号" prop="ptrNum">
            <el-input v-model="form.ptrNum" placeholder="请输入记录序号" />
        </el-form-item>
        <el-form-item label="成绩" prop="ptrScore">
            <el-input v-model="form.ptrScore" placeholder="请输入成绩" />
        </el-form-item>
        <el-form-item label="权重分" prop="ptrWscore">
            <el-input v-model="form.ptrWscore" placeholder="请输入权重分" />
        </el-form-item>
        <el-form-item label="测试日期" prop="ptrDate">
            <el-date-picker clearable
```

```
                v-model="form.ptrDate"
                type="date"
                value-format="yyyy-MM-dd"
                placeholder="请选择测试日期">
              </el-date-picker>
            </el-form-item>
          </el-form>
          <div slot="footer" class="dialog-footer">
            <el-button type="primary" @click="submitForm">确 定</el-button>
            <el-button @click="cancel">取 消</el-button>
          </div>
        </el-dialog>
    </div>
</template>

<script>
import { listPtrecord, getPtrecord, delPtrecord, addPtrecord, updatePtrecord } from
"@/api/ptrecordManager/ptrecord";

export default {
    name: "Ptrecord",
    data() {
        return {
            // 遮罩层
            loading: true,
            // 选中数组
            ids: [],
```

```
// 非单个禁用
single: true,
// 非多个禁用
multiple: true,
// 显示搜索条件
showSearch: true,
// 总条数
total: 0,
// 体测记录管理表格数据
ptrecordList: [],
// 弹出层标题
title: "",
// 是否显示弹出层
open: false,
// 查询参数
queryParams: {
    pageNum: 1,
    pageSize: 10,
    macId: null,
    adminId: null,
    ptId: null,
    stuId: null,
    ptrNum: null,
    ptrScore: null,
    ptrWscore: null,
    ptrDate: null
},
```

```
      // 表单参数
      form: {},
      // 表单校验
      rules: {
      }
    };
},
created() {
    this.getList();
},
methods: {
    /** 查询体测记录管理列表 */
    getList() {
      this.loading = true;
      listPtrecord(this.queryParams).then(response => {
        this.ptrecordList = response.rows;
        this.total = response.total;
        this.loading = false;
      });
    },
    // 取消按钮
    cancel() {
      this.open = false;
      this.reset();
    },
    // 表单重置
    reset() {
```

```
    this.form = {
        ptrId: null,
        macId: null,
        adminId: null,
        ptId: null,
        stuId: null,
        ptrNum: null,
        ptrScore: null,
        ptrWscore: null,
        ptrDate: null
    };
    this.resetForm("form");
},
/** 搜索按钮操作 */
handleQuery() {
    this.queryParams.pageNum = 1;
    this.getList();
},
/** 重置按钮操作 */
resetQuery() {
    this.resetForm("queryForm");
    this.handleQuery();
},
// 多选框选中数据
handleSelectionChange(selection) {
    this.ids = selection.map(item => item.ptrId)
    this.single = selection.length!==1
```

```
    this.multiple = !selection.length
},
/** 新增按钮操作 */
handleAdd() {
    this.reset();
    this.open = true;
    this.title = "添加体测记录管理";
},
/** 修改按钮操作 */
handleUpdate(row) {
    this.reset();
    const ptrId = row.ptrId || this.ids
    getPtrecord(ptrId).then(response => {
        this.form = response.data;
        this.open = true;
        this.title = "修改体测记录管理";
    });
},
/** 提交按钮 */
submitForm() {
    this.$refs["form"].validate(valid => {
        if (valid) {
            if (this.form.ptrId != null) {
                updatePtrecord(this.form).then(response => {
                    this.$modal.msgSuccess("修改成功");
                    this.open = false;
                    this.getList();
```

```
                });
            } else {
                addPtrecord(this.form).then(response => {
                    this.$modal.msgSuccess("新增成功");
                    this.open = false;
                    this.getList();
                });
            }
        }
    });
},
/** 删除按钮操作 */
handleDelete(row) {
    const ptrIds = row.ptrId || this.ids;
    this.$modal.confirm('是否确认删除体测记录管理编号为"' + ptrIds + '"的数据项？').then(function() {
        return delPtrecord(ptrIds);
    }).then(() => {
        this.getList();
        this.$modal.msgSuccess("删除成功");
    }).catch(() => {});
},
/** 导出按钮操作 */
handleExport() {
    this.download('ptrecordManager/ptrecord/export', {
        ...this.queryParams
    }, `ptrecord_${new Date().getTime()}.xlsx`)
```

```
    }
  }
};
</script>
```

src/view/ptscoreManager/ptscore/index.vue

```
<template>
  <div class="app-container">
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-show="showSearch" label-width="68px">
      <el-form-item label="学生 id" prop="stuId">
        <el-input
          v-model="queryParams.stuId"
          placeholder="请输入学生 id"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="成绩" prop="ptscoreScore">
        <el-input
          v-model="queryParams.ptscoreScore"
          placeholder="请输入成绩"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="起始日期" prop="ptscoreSatart">
```

```
        <el-date-picker clearable
          v-model="queryParams.ptscoreSatart"
          type="date"
          value-format="yyyy-MM-dd"
          placeholder="请选择起始日期">
        </el-date-picker>
      </el-form-item>
      <el-form-item label="结束日期" prop="ptscoreEnd">
        <el-date-picker clearable
          v-model="queryParams.ptscoreEnd"
          type="date"
          value-format="yyyy-MM-dd"
          placeholder="请选择结束日期">
        </el-date-picker>
      </el-form-item>
      <el-form-item label="是否缺项" prop="ptscoreLack">
        <el-input
          v-model="queryParams.ptscoreLack"
          placeholder="请输入是否缺项"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item>
        <el-button type="primary" icon="el-icon-search" size="mini"
@click="handleQuery">搜索</el-button>
        <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置
```

```
</el-button>
        </el-form-item>
      </el-form>

      <el-row :gutter="10" class="mb8">
        <el-col :span="1.5">
          <el-button
            type="primary"
            plain
            icon="el-icon-plus"
            size="mini"
            @click="handleAdd"
            v-hasPermi="['ptscoreManager:ptscore:add']"
          >新增</el-button>
        </el-col>
        <el-col :span="1.5">
          <el-button
            type="success"
            plain
            icon="el-icon-edit"
            size="mini"
            :disabled="single"
            @click="handleUpdate"
            v-hasPermi="['ptscoreManager:ptscore:edit']"
          >修改</el-button>
        </el-col>
        <el-col :span="1.5">
```

```
            <el-button
              type="danger"
              plain
              icon="el-icon-delete"
              size="mini"
              :disabled="multiple"
              @click="handleDelete"
              v-hasPermi="['ptscoreManager:ptscore:remove']"
            >删除</el-button>
          </el-col>
          <el-col :span="1.5">
            <el-button
              type="warning"
              plain
              icon="el-icon-download"
              size="mini"
              @click="handleExport"
              v-hasPermi="['ptscoreManager:ptscore:export']"
            >导出</el-button>
          </el-col>
          <right-toolbar :showSearch.sync="showSearch" @queryTable="getList"></right-toolbar>
        </el-row>

        <el-table v-loading="loading" :data="ptscoreList" @selection-change="handleSelectionChange">
          <el-table-column type="selection" width="55" align="center" />
```

```
        <el-table-column label="成绩 id" align="center" prop="ptscoreId" />
        <el-table-column label="学生 id" align="center" prop="stuId" />
        <el-table-column label="成绩" align="center" prop="ptscoreScore" />
        <el-table-column label="起始日期" align="center" prop="ptscoreSatart" width="180">
          <template slot-scope="scope">
            <span>{{ parseTime(scope.row.ptscoreSatart, '{y}-{m}-{d}') }}</span>
          </template>
        </el-table-column>
        <el-table-column label="结束日期" align="center" prop="ptscoreEnd" width="180">
          <template slot-scope="scope">
            <span>{{ parseTime(scope.row.ptscoreEnd, '{y}-{m}-{d}') }}</span>
          </template>
        </el-table-column>
        <el-table-column label="是否缺项" align="center" prop="ptscoreLack" />
        <el-table-column label="操作" align="center" class-name="small-padding fixed-width">
          <template slot-scope="scope">
            <el-button
              size="mini"
              type="text"
              icon="el-icon-edit"
              @click="handleUpdate(scope.row)"
              v-hasPermi="['ptscoreManager:ptscore:edit']"
            >修改</el-button>
            <el-button
              size="mini"
```

```
            type="text"
            icon="el-icon-delete"
            @click="handleDelete(scope.row)"
            v-hasPermi="['ptscoreManager:ptscore:remove']"
          >删除</el-button>
        </template>
      </el-table-column>
    </el-table>

    <pagination
      v-show="total>0"
      :total="total"
      :page.sync="queryParams.pageNum"
      :limit.sync="queryParams.pageSize"
      @pagination="getList"
    />

    <!-- 添加或修改学生分数管理对话框 -->
    <el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
      <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="学生 id" prop="stuId">
          <el-input v-model="form.stuId" placeholder="请输入学生 id" />
        </el-form-item>
        <el-form-item label="成绩" prop="ptscoreScore">
          <el-input v-model="form.ptscoreScore" placeholder="请输入成绩" />
        </el-form-item>
        <el-form-item label="起始日期" prop="ptscoreSatart">
```

```
            <el-date-picker clearable
              v-model="form.ptscoreSatart"
              type="date"
              value-format="yyyy-MM-dd"
              placeholder="请选择起始日期">
            </el-date-picker>
          </el-form-item>
          <el-form-item label="结束日期" prop="ptscoreEnd">
            <el-date-picker clearable
              v-model="form.ptscoreEnd"
              type="date"
              value-format="yyyy-MM-dd"
              placeholder="请选择结束日期">
            </el-date-picker>
          </el-form-item>
          <el-form-item label="是否缺项" prop="ptscoreLack">
            <el-input v-model="form.ptscoreLack" placeholder="请输入是否缺项" />
          </el-form-item>
        </el-form>
        <div slot="footer" class="dialog-footer">
          <el-button type="primary" @click="submitForm">确  定</el-button>
          <el-button @click="cancel">取  消</el-button>
        </div>
      </el-dialog>
    </div>
</template>
```

```
<script>
import { listPtscore, getPtscore, delPtscore, addPtscore, updatePtscore } from "@/api/ptscoreManager/ptscore";

export default {
  name: "Ptscore",
  data() {
    return {
      // 遮罩层
      loading: true,
      // 选中数组
      ids: [],
      // 非单个禁用
      single: true,
      // 非多个禁用
      multiple: true,
      // 显示搜索条件
      showSearch: true,
      // 总条数
      total: 0,
      // 学生分数管理表格数据
      ptscoreList: [],
      // 弹出层标题
      title: "",
      // 是否显示弹出层
      open: false,
      // 查询参数
```

```
      queryParams: {
        pageNum: 1,
        pageSize: 10,
        stuId: null,
        ptscoreScore: null,
        ptscoreSatart: null,
        ptscoreEnd: null,
        ptscoreLack: null
      },
      // 表单参数
      form: {},
      // 表单校验
      rules: {
      }
    };
  },
  created() {
    this.getList();
  },
  methods: {
    /** 查询学生分数管理列表 */
    getList() {
      this.loading = true;
      listPtscore(this.queryParams).then(response => {
        this.ptscoreList = response.rows;
        this.total = response.total;
        this.loading = false;
```

```
    });
},
// 取消按钮
cancel() {
    this.open = false;
    this.reset();
},
// 表单重置
reset() {
    this.form = {
        ptscoreId: null,
        stuId: null,
        ptscoreScore: null,
        ptscoreSatart: null,
        ptscoreEnd: null,
        ptscoreLack: null
    };
    this.resetForm("form");
},
/** 搜索按钮操作 */
handleQuery() {
    this.queryParams.pageNum = 1;
    this.getList();
},
/** 重置按钮操作 */
resetQuery() {
    this.resetForm("queryForm");
```

```
    this.handleQuery();
  },
  // 多选框选中数据
  handleSelectionChange(selection) {
    this.ids = selection.map(item => item.ptscoreId)
    this.single = selection.length!==1
    this.multiple = !selection.length
  },
  /** 新增按钮操作 */
  handleAdd() {
    this.reset();
    this.open = true;
    this.title = "添加学生分数管理";
  },
  /** 修改按钮操作 */
  handleUpdate(row) {
    this.reset();
    const ptscoreId = row.ptscoreId || this.ids
    getPtscore(ptscoreId).then(response => {
      this.form = response.data;
      this.open = true;
      this.title = "修改学生分数管理";
    });
  },
  /** 提交按钮 */
  submitForm() {
    this.$refs["form"].validate(valid => {
```

```
        if (valid) {
          if (this.form.ptscoreId != null) {
            updatePtscore(this.form).then(response => {
              this.$modal.msgSuccess("修改成功");
              this.open = false;
              this.getList();
            });
          } else {
            addPtscore(this.form).then(response => {
              this.$modal.msgSuccess("新增成功");
              this.open = false;
              this.getList();
            });
          }
        }
      });
    },
    /** 删除按钮操作 */
    handleDelete(row) {
      const ptscoreIds = row.ptscoreId || this.ids;
      this.$modal.confirm('是否确认删除学生分数管理编号为"' + ptscoreIds + '"的数据项？').then(function() {
        return delPtscore(ptscoreIds);
      }).then(() => {
        this.getList();
        this.$modal.msgSuccess("删除成功");
      }).catch(() => {});
```

```
    },
    /** 导出按钮操作 */
    handleExport() {
      this.download('ptscoreManager/ptscore/export', {
        ...this.queryParams
      }, `ptscore_${new Date().getTime()}.xlsx`)
    }
  }
};
</script>
```

src/view/ptstandManager/ptstandManager/index.vue

```
<template>
  <div class="app-container">
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-show="showSearch" label-width="68px">
      <el-form-item label="体测项目 id" prop="ptId">
        <el-input
          v-model="queryParams.ptId"
          placeholder="请输入体测项目 id"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="体测标准编号" prop="standNum">
        <el-input
          v-model="queryParams.standNum"
```

```
        placeholder="请输入体测标准编号"
        clearable
        @keyup.enter.native="handleQuery"
      />
    </el-form-item>
    <el-form-item label="标准等级" prop="standLevel">
      <el-input
        v-model="queryParams.standLevel"
        placeholder="请输入标准等级"
        clearable
        @keyup.enter.native="handleQuery"
      />
    </el-form-item>
    <el-form-item label="得分" prop="standScore">
      <el-input
        v-model="queryParams.standScore"
        placeholder="请输入得分"
        clearable
        @keyup.enter.native="handleQuery"
      />
    </el-form-item>
    <el-form-item label="年级" prop="standGrade">
      <el-input
        v-model="queryParams.standGrade"
        placeholder="请输入年级"
        clearable
        @keyup.enter.native="handleQuery"
```

```
      />
    </el-form-item>
    <el-form-item label="性别" prop="standGender">
      <el-select v-model="queryParams.standGender" placeholder="请选择性别"
clearable>
        <el-option
          v-for="dict in dict.type.sys_user_sex"
          :key="dict.value"
          :label="dict.label"
          :value="dict.value"
        />
      </el-select>
    </el-form-item>
    <el-form-item label="最低成绩" prop="standMin">
      <el-input
        v-model="queryParams.standMin"
        placeholder="请输入最低成绩"
        clearable
        @keyup.enter.native="handleQuery"
      />
    </el-form-item>
    <el-form-item label="最高成绩" prop="standMax">
      <el-input
        v-model="queryParams.standMax"
        placeholder="请输入最高成绩"
        clearable
        @keyup.enter.native="handleQuery"
```

```
        />
      </el-form-item>
      <el-form-item>
        <el-button type="primary" icon="el-icon-search" size="mini" @click="handleQuery">搜索</el-button>
        <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置</el-button>
      </el-form-item>
    </el-form>

    <el-row :gutter="10" class="mb8">
      <el-col :span="1.5">
        <el-button
          type="primary"
          plain
          icon="el-icon-plus"
          size="mini"
          @click="handleAdd"
          v-hasPermi="['ptstandManager:ptstandManager:add']"
        >新增</el-button>
      </el-col>
      <el-col :span="1.5">
        <el-button
          type="success"
          plain
          icon="el-icon-edit"
          size="mini"
```

```
        :disabled="single"
        @click="handleUpdate"
        v-hasPermi="['ptstandManager:ptstandManager:edit']"
      >修改</el-button>
    </el-col>
    <el-col :span="1.5">
      <el-button
        type="danger"
        plain
        icon="el-icon-delete"
        size="mini"
        :disabled="multiple"
        @click="handleDelete"
        v-hasPermi="['ptstandManager:ptstandManager:remove']"
      >删除</el-button>
    </el-col>
    <el-col :span="1.5">
      <el-button
        type="warning"
        plain
        icon="el-icon-download"
        size="mini"
        @click="handleExport"
        v-hasPermi="['ptstandManager:ptstandManager:export']"
      >导出</el-button>
    </el-col>
    <right-toolbar :showSearch.sync="showSearch"
```

```
@queryTable="getList"></right-toolbar>
    </el-row>

    <el-table v-loading="loading" :data="ptstandManagerList" @selection-change="handleSelectionChange">
      <el-table-column type="selection" width="55" align="center" />
      <el-table-column label="体测标准 id" align="center" prop="standId" />
      <el-table-column label="体测项目 id" align="center" prop="ptId" />
      <el-table-column label="体测标准编号" align="center" prop="standNum" />
      <el-table-column label="标准等级" align="center" prop="standLevel" />
      <el-table-column label="得分" align="center" prop="standScore" />
      <el-table-column label="年级" align="center" prop="standGrade" />
      <el-table-column label="性别" align="center" prop="standGender">
        <template slot-scope="scope">
          <dict-tag :options="dict.type.sys_user_sex" :value="scope.row.standGender"/>
        </template>
      </el-table-column>
      <el-table-column label="最低成绩" align="center" prop="standMin" />
      <el-table-column label="最高成绩" align="center" prop="standMax" />
      <el-table-column label="操作" align="center" class-name="small-padding fixed-width">
        <template slot-scope="scope">
          <el-button
            size="mini"
            type="text"
            icon="el-icon-edit"
            @click="handleUpdate(scope.row)"
```

```
            v-hasPermi="['ptstandManager:ptstandManager:edit']"
          >修改</el-button>
          <el-button
            size="mini"
            type="text"
            icon="el-icon-delete"
            @click="handleDelete(scope.row)"
            v-hasPermi="['ptstandManager:ptstandManager:remove']"
          >删除</el-button>
        </template>
      </el-table-column>
    </el-table>

    <pagination
      v-show="total>0"
      :total="total"
      :page.sync="queryParams.pageNum"
      :limit.sync="queryParams.pageSize"
      @pagination="getList"
    />

    <!-- 添加或修改体测标准对话框 -->
    <el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
      <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="体测项目 id" prop="ptId">
          <el-input v-model="form.ptId" placeholder="请输入体测项目 id" />
        </el-form-item>
```

```
            <el-form-item label="体测标准编号" prop="standNum">
              <el-input v-model="form.standNum" placeholder="请输入体测标准编号"
/>
            </el-form-item>
            <el-form-item label="标准等级" prop="standLevel">
              <el-input v-model="form.standLevel" placeholder="请输入标准等级" />
            </el-form-item>
            <el-form-item label="得分" prop="standScore">
              <el-input v-model="form.standScore" placeholder="请输入得分" />
            </el-form-item>
            <el-form-item label="年级" prop="standGrade">
              <el-input v-model="form.standGrade" placeholder="请输入年级" />
            </el-form-item>
            <el-form-item label="性别" prop="standGender">
              <el-radio-group v-model="form.standGender">
                <el-radio
                  v-for="dict in dict.type.sys_user_sex"
                  :key="dict.value"
                  :label="dict.value"
                >{{dict.label}}</el-radio>
              </el-radio-group>
            </el-form-item>
            <el-form-item label="最低成绩" prop="standMin">
              <el-input v-model="form.standMin" placeholder="请输入最低成绩" />
            </el-form-item>
            <el-form-item label="最高成绩" prop="standMax">
              <el-input v-model="form.standMax" placeholder="请输入最高成绩" />
```

```
            </el-form-item>
          </el-form>
          <div slot="footer" class="dialog-footer">
            <el-button type="primary" @click="submitForm">确 定</el-button>
            <el-button @click="cancel">取 消</el-button>
          </div>
        </el-dialog>
    </div>
</template>

<script>
import { listPtstandManager, getPtstandManager, delPtstandManager,
addPtstandManager, updatePtstandManager } from
"@/api/ptstandManager/ptstandManager";

export default {
    name: "PtstandManager",
    dicts: ['sys_user_sex'],
    data() {
        return {
            // 遮罩层
            loading: true,
            // 选中数组
            ids: [],
            // 非单个禁用
            single: true,
            // 非多个禁用
```

```
multiple: true,
// 显示搜索条件
showSearch: true,
// 总条数
total: 0,
// 体测标准表格数据
ptstandManagerList: [],
// 弹出层标题
title: "",
// 是否显示弹出层
open: false,
// 查询参数
queryParams: {
    pageNum: 1,
    pageSize: 10,
    ptId: null,
    standNum: null,
    standLevel: null,
    standScore: null,
    standGrade: null,
    standGender: null,
    standMin: null,
    standMax: null
},
// 表单参数
form: {},
// 表单校验
```

```
        rules: {
          standId: [
            { required: true, message: "体测标准 id 不能为空", trigger: "blur" }
          ],
          ptId: [
            { required: true, message: "体测项目 id 不能为空", trigger: "blur" }
          ],
          standNum: [
            { required: true, message: "体测标准编号不能为空", trigger: "blur" }
          ],
          standLevel: [
            { required: true, message: "标准等级不能为空", trigger: "blur" }
          ],
          standScore: [
            { required: true, message: "得分不能为空", trigger: "blur" }
          ],
          standMin: [
            { required: true, message: "最低成绩不能为空", trigger: "blur" }
          ],
          standMax: [
            { required: true, message: "最高成绩不能为空", trigger: "blur" }
          ]
        }
      };
    },
    created() {
      this.getList();
```

```
},
methods: {
  /** 查询体测标准列表 */
  getList() {
    this.loading = true;
    listPtstandManager(this.queryParams).then(response => {
      this.ptstandManagerList = response.rows;
      this.total = response.total;
      this.loading = false;
    });
  },
  // 取消按钮
  cancel() {
    this.open = false;
    this.reset();
  },
  // 表单重置
  reset() {
    this.form = {
      standId: null,
      ptId: null,
      standNum: null,
      standLevel: null,
      standScore: null,
      standGrade: null,
      standGender: null,
      standMin: null,
```

```
        standMax: null
      };
      this.resetForm("form");
    },
    /** 搜索按钮操作 */
    handleQuery() {
      this.queryParams.pageNum = 1;
      this.getList();
    },
    /** 重置按钮操作 */
    resetQuery() {
      this.resetForm("queryForm");
      this.handleQuery();
    },
    // 多选框选中数据
    handleSelectionChange(selection) {
      this.ids = selection.map(item => item.standId)
      this.single = selection.length!==1
      this.multiple = !selection.length
    },
    /** 新增按钮操作 */
    handleAdd() {
      this.reset();
      this.open = true;
      this.title = "添加体测标准";
    },
    /** 修改按钮操作 */
```

```
handleUpdate(row) {
    this.reset();
    const standId = row.standId || this.ids
    getPtstandManager(standId).then(response => {
        this.form = response.data;
        this.open = true;
        this.title = "修改体测标准";
    });
},
/** 提交按钮 */
submitForm() {
    this.$refs["form"].validate(valid => {
        if (valid) {
            if (this.form.standId != null) {
                updatePtstandManager(this.form).then(response => {
                    this.$modal.msgSuccess("修改成功");
                    this.open = false;
                    this.getList();
                });
            } else {
                addPtstandManager(this.form).then(response => {
                    this.$modal.msgSuccess("新增成功");
                    this.open = false;
                    this.getList();
                });
            }
        }
```

```
        });
      },
      /** 删除按钮操作 */
      handleDelete(row) {
        const standIds = row.standId || this.ids;
        this.$modal.confirm('是否确认删除体测标准编号为"' + standIds + '"的数据项？
').then(function() {
          return delPtstandManager(standIds);
        }).then(() => {
          this.getList();
          this.$modal.msgSuccess("删除成功");
        }).catch(() => {});
      },
      /** 导出按钮操作 */
      handleExport() {
        this.download('ptstandManager/ptstandManager/export', {
          ...this.queryParams
        }, `ptstandManager_${new Date().getTime()}.xlsx`)
      }
    }
};
</script>
```

src/view/stuManager/stu/index.vue

```
<template>
  <div class="app-container">
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-
```

```
show="showSearch" label-width="68px">
        <el-form-item label="学号" prop="stuNum">
          <el-input
            v-model="queryParams.stuNum"
            placeholder="请输入学号"
            clearable
            @keyup.enter.native="handleQuery"
          />
        </el-form-item>
        <el-form-item label="姓名" prop="stuName">
          <el-input
            v-model="queryParams.stuName"
            placeholder="请输入姓名"
            clearable
            @keyup.enter.native="handleQuery"
          />
        </el-form-item>
        <el-form-item label="性别" prop="stuSex">
          <el-select v-model="queryParams.stuSex" placeholder="请选择性别"
clearable>
            <el-option
              v-for="dict in dict.type.sys_user_sex"
              :key="dict.value"
              :label="dict.label"
              :value="dict.value"
            />
          </el-select>
```

```
</el-form-item>
<el-form-item label="学院" prop="college">
  <el-input
    v-model="queryParams.college"
    placeholder="请输入学院"
    clearable
    @keyup.enter.native="handleQuery"
  />
</el-form-item>
<el-form-item label="专业" prop="major">
  <el-input
    v-model="queryParams.major"
    placeholder="请输入专业"
    clearable
    @keyup.enter.native="handleQuery"
  />
</el-form-item>
<el-form-item label="班级" prop="stuClass">
  <el-input
    v-model="queryParams.stuClass"
    placeholder="请输入班级"
    clearable
    @keyup.enter.native="handleQuery"
  />
</el-form-item>
<el-form-item label="电话" prop="stuPhone">
  <el-input
```

```
            v-model="queryParams.stuPhone"
            placeholder="请输入电话"
            clearable
            @keyup.enter.native="handleQuery"
          />
        </el-form-item>
        <el-form-item label="电子邮件" prop="stuEmail">
          <el-input
            v-model="queryParams.stuEmail"
            placeholder="请输入电子邮件"
            clearable
            @keyup.enter.native="handleQuery"
          />
        </el-form-item>
        <el-form-item>
          <el-button type="primary" icon="el-icon-search" size="mini"
@click="handleQuery">搜索</el-button>
          <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置
</el-button>
        </el-form-item>
      </el-form>

      <el-row :gutter="10" class="mb8">
        <el-col :span="1.5">
          <el-button
            type="primary"
            plain
```

```
        icon="el-icon-plus"
        size="mini"
        @click="handleAdd"
        v-hasPermi="['stuManager:stu:add']"
    >新增</el-button>
</el-col>
<el-col :span="1.5">
    <el-button
        type="success"
        plain
        icon="el-icon-edit"
        size="mini"
        :disabled="single"
        @click="handleUpdate"
        v-hasPermi="['stuManager:stu:edit']"
    >修改</el-button>
</el-col>
<el-col :span="1.5">
    <el-button
        type="danger"
        plain
        icon="el-icon-delete"
        size="mini"
        :disabled="multiple"
        @click="handleDelete"
        v-hasPermi="['stuManager:stu:remove']"
    >删除</el-button>
```

```
      </el-col>
      <el-col :span="1.5">
        <el-button
          type="warning"
          plain
          icon="el-icon-download"
          size="mini"
          @click="handleExport"
          v-hasPermi="['stuManager:stu:export']"
        >导出</el-button>
      </el-col>
      <right-toolbar :showSearch.sync="showSearch" @queryTable="getList"></right-toolbar>
    </el-row>

    <el-table v-loading="loading" :data="stuList" @selection-change="handleSelectionChange">
      <el-table-column type="selection" width="55" align="center" />
      <el-table-column label="学生 id" align="center" prop="stuId" />
      <el-table-column label="学号" align="center" prop="stuNum" />
      <el-table-column label="姓名" align="center" prop="stuName" />
      <el-table-column label="性别" align="center" prop="stuSex">
        <template slot-scope="scope">
          <dict-tag :options="dict.type.sys_user_sex" :value="scope.row.stuSex"/>
        </template>
      </el-table-column>
      <el-table-column label="学院" align="center" prop="college" />
```

```
        <el-table-column label="专业" align="center" prop="major" />
        <el-table-column label="班级" align="center" prop="stuClass" />
        <el-table-column label="电话" align="center" prop="stuPhone" />
        <el-table-column label="电子邮件" align="center" prop="stuEmail" />
        <el-table-column label="头像" align="center" prop="stuAvatar" width="100">
          <template slot-scope="scope">
            <image-preview :src="scope.row.stuAvatar" :width="50" :height="50"/>
          </template>
        </el-table-column>
        <el-table-column label="操作" align="center" class-name="small-padding
fixed-width">
          <template slot-scope="scope">
            <el-button
              size="mini"
              type="text"
              icon="el-icon-edit"
              @click="handleUpdate(scope.row)"
              v-hasPermi="['stuManager:stu:edit']"
            >修改</el-button>
            <el-button
              size="mini"
              type="text"
              icon="el-icon-delete"
              @click="handleDelete(scope.row)"
              v-hasPermi="['stuManager:stu:remove']"
            >删除</el-button>
          </template>
```

```
    </el-table-column>
</el-table>

<pagination
    v-show="total>0"
    :total="total"
    :page.sync="queryParams.pageNum"
    :limit.sync="queryParams.pageSize"
    @pagination="getList"
/>

<!-- 添加或修改学生信息管理对话框 -->
<el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
    <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="学校 id" prop="schId">
            <el-input v-model="form.schId" placeholder="请输入学校 id" />
        </el-form-item>
        <el-form-item label="学号" prop="stuNum">
            <el-input v-model="form.stuNum" placeholder="请输入学号" />
        </el-form-item>
        <el-form-item label="姓名" prop="stuName">
            <el-input v-model="form.stuName" placeholder="请输入姓名" />
        </el-form-item>
        <el-form-item label="性别" prop="stuSex">
            <el-radio-group v-model="form.stuSex">
                <el-radio
                    v-for="dict in dict.type.sys_user_sex"
```

```
            :key="dict.value"
            :label="dict.value"
          >{{dict.label}}</el-radio>
        </el-radio-group>
      </el-form-item>
      <el-form-item label="学院" prop="college">
        <el-input v-model="form.college" placeholder="请输入学院" />
      </el-form-item>
      <el-form-item label="专业" prop="major">
        <el-input v-model="form.major" placeholder="请输入专业" />
      </el-form-item>
      <el-form-item label="班级" prop="stuClass">
        <el-input v-model="form.stuClass" placeholder="请输入班级" />
      </el-form-item>
      <el-form-item label="密码" prop="stuPwd">
        <el-input v-model="form.stuPwd" placeholder="请输入密码" />
      </el-form-item>
      <el-form-item label="电话" prop="stuPhone">
        <el-input v-model="form.stuPhone" placeholder="请输入电话" />
      </el-form-item>
      <el-form-item label="电子邮件" prop="stuEmail">
        <el-input v-model="form.stuEmail" placeholder="请输入电子邮件" />
      </el-form-item>
      <el-form-item label="头像" prop="stuAvatar">
        <image-upload v-model="form.stuAvatar"/>
      </el-form-item>
    </el-form>
```

```
        <div slot="footer" class="dialog-footer">
          <el-button type="primary" @click="submitForm">确 定</el-button>
          <el-button @click="cancel">取 消</el-button>
        </div>
      </el-dialog>
    </div>
</template>

<script>
import { listStu, getStu, delStu, addStu, updateStu } from "@/api/stuManager/stu";

export default {
    name: "Stu",
    dicts: ['sys_user_sex'],
    data() {
      return {
        // 遮罩层
        loading: true,
        // 选中数组
        ids: [],
        // 非单个禁用
        single: true,
        // 非多个禁用
        multiple: true,
        // 显示搜索条件
        showSearch: true,
        // 总条数
```

```
total: 0,
// 学生信息管理表格数据
stuList: [],
// 弹出层标题
title: "",
// 是否显示弹出层
open: false,
// 查询参数
queryParams: {
  pageNum: 1,
  pageSize: 10,
  stuNum: null,
  stuName: null,
  stuSex: null,
  college: null,
  major: null,
  stuClass: null,
  stuPhone: null,
  stuEmail: null,
},
// 表单参数
form: {},
// 表单校验
rules: {
  stuId: [
    { required: true, message: "学生 id 不能为空", trigger: "blur" }
  ],
```

```
        schId: [
          { required: true, message: "学校 id 不能为空", trigger: "blur" }
        ],
        stuNum: [
          { required: true, message: "学号不能为空", trigger: "blur" }
        ],
        stuName: [
          { required: true, message: "姓名不能为空", trigger: "blur" }
        ],
        stuSex: [
          { required: true, message: "性别不能为空", trigger: "change" }
        ],
        college: [
          { required: true, message: "学院不能为空", trigger: "blur" }
        ],
        major: [
          { required: true, message: "专业不能为空", trigger: "blur" }
        ],
        stuClass: [
          { required: true, message: "班级不能为空", trigger: "blur" }
        ],
        stuPwd: [
          { required: true, message: "密码不能为空", trigger: "blur" }
        ],
      }
    };
  },
```

```
created() {
    this.getList();
},
methods: {
    /** 查询学生信息管理列表 */
    getList() {
        this.loading = true;
        listStu(this.queryParams).then(response => {
            this.stuList = response.rows;
            this.total = response.total;
            this.loading = false;
        });
    },
    // 取消按钮
    cancel() {
        this.open = false;
        this.reset();
    },
    // 表单重置
    reset() {
        this.form = {
            stuId: null,
            schId: null,
            stuNum: null,
            stuName: null,
            stuSex: null,
            college: null,
```

```
        major: null,
        stuClass: null,
        stuPwd: null,
        stuPhone: null,
        stuEmail: null,
        stuAvatar: null
    };
    this.resetForm("form");
},
/** 搜索按钮操作 */
handleQuery() {
    this.queryParams.pageNum = 1;
    this.getList();
},
/** 重置按钮操作 */
resetQuery() {
    this.resetForm("queryForm");
    this.handleQuery();
},
// 多选框选中数据
handleSelectionChange(selection) {
    this.ids = selection.map(item => item.stuId)
    this.single = selection.length!==1
    this.multiple = !selection.length
},
/** 新增按钮操作 */
handleAdd() {
```

```
    this.reset();
    this.open = true;
    this.title = "添加学生信息管理";
},
/** 修改按钮操作 */
handleUpdate(row) {
    this.reset();
    const stuId = row.stuId || this.ids
    getStu(stuId).then(response => {
        this.form = response.data;
        this.open = true;
        this.title = "修改学生信息管理";
    });
},
/** 提交按钮 */
submitForm() {
    this.$refs["form"].validate(valid => {
        if (valid) {
            if (this.form.stuId != null) {
                updateStu(this.form).then(response => {
                    this.$modal.msgSuccess("修改成功");
                    this.open = false;
                    this.getList();
                });
            } else {
                addStu(this.form).then(response => {
                    this.$modal.msgSuccess("新增成功");
```

```
                    this.open = false;
                    this.getList();
                });
            }
        }
    });
},
/** 删除按钮操作 */
handleDelete(row) {
    const stuIds = row.stuId || this.ids;
    this.$modal.confirm('是否确认删除学生信息管理编号为"' + stuIds + '"的数据项？').then(function() {
        return delStu(stuIds);
    }).then(() => {
        this.getList();
        this.$modal.msgSuccess("删除成功");
    }).catch(() => {});
},
/** 导出按钮操作 */
handleExport() {
    this.download('stuManager/stu/export', {
        ...this.queryParams
    }, `stu_${new Date().getTime()}.xlsx`)
}
  }
};
</script>
```

src/view/teaManager/teaManager/index.vue

```
<template>
  <div class="app-container">
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-show="showSearch" label-width="68px">
      <el-form-item label="教师工号" prop="teaNum">
        <el-input
          v-model="queryParams.teaNum"
          placeholder="请输入教师工号"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="教师名称" prop="teaName">
        <el-input
          v-model="queryParams.teaName"
          placeholder="请输入教师名称"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="教师性别" prop="teaSex">
        <el-select v-model="queryParams.teaSex" placeholder="请选择教师性别" clearable>
          <el-option
            v-for="dict in dict.type.sys_user_sex"
```

```
            :key="dict.value"
            :label="dict.label"
            :value="dict.value"
          />
        </el-select>
      </el-form-item>
      <el-form-item label="教师电话" prop="teaPhone">
        <el-input
          v-model="queryParams.teaPhone"
          placeholder="请输入教师电话"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item label="教师邮箱" prop="teaEmail">
        <el-input
          v-model="queryParams.teaEmail"
          placeholder="请输入教师邮箱"
          clearable
          @keyup.enter.native="handleQuery"
        />
      </el-form-item>
      <el-form-item>
        <el-button type="primary" icon="el-icon-search" size="mini"
@click="handleQuery">搜索</el-button>
        <el-button icon="el-icon-refresh" size="mini" @click="resetQuery">重置
</el-button>
```

```
    </el-form-item>
</el-form>

<el-row :gutter="10" class="mb8">
    <el-col :span="1.5">
        <el-button
            type="primary"
            plain
            icon="el-icon-plus"
            size="mini"
            @click="handleAdd"
            v-hasPermi="['teaManager:teaManager:add']"
        >新增</el-button>
    </el-col>
    <el-col :span="1.5">
        <el-button
            type="success"
            plain
            icon="el-icon-edit"
            size="mini"
            :disabled="single"
            @click="handleUpdate"
            v-hasPermi="['teaManager:teaManager:edit']"
        >修改</el-button>
    </el-col>
    <el-col :span="1.5">
        <el-button
```

```
            type="danger"
            plain
            icon="el-icon-delete"
            size="mini"
            :disabled="multiple"
            @click="handleDelete"
            v-hasPermi="['teaManager:teaManager:remove']"
          >删除</el-button>
        </el-col>
        <el-col :span="1.5">
          <el-button
            type="warning"
            plain
            icon="el-icon-download"
            size="mini"
            @click="handleExport"
            v-hasPermi="['teaManager:teaManager:export']"
          >导出</el-button>
        </el-col>
        <right-toolbar :showSearch.sync="showSearch" @queryTable="getList"></right-toolbar>
      </el-row>

      <el-table v-loading="loading" :data="teaManagerList" @selection-change="handleSelectionChange">
        <el-table-column type="selection" width="55" align="center" />
        <el-table-column label="教师 id" align="center" prop="teaId" />
```

```
        <el-table-column label="教师工号" align="center" prop="teaNum" />
        <el-table-column label="教师名称" align="center" prop="teaName" />
        <el-table-column label="教师性别" align="center" prop="teaSex">
          <template slot-scope="scope">
            <dict-tag :options="dict.type.sys_user_sex" :value="scope.row.teaSex"/>
          </template>
        </el-table-column>
        <el-table-column label="教师电话" align="center" prop="teaPhone" />
        <el-table-column label="教师邮箱" align="center" prop="teaEmail" />
        <el-table-column label="教师头像" align="center" prop="teaAvatar"
width="100">
          <template slot-scope="scope">
            <image-preview :src="scope.row.teaAvatar" :width="50" :height="50"/>
          </template>
        </el-table-column>
        <el-table-column label="操作" align="center" class-name="small-padding
fixed-width">
          <template slot-scope="scope">
            <el-button
              size="mini"
              type="text"
              icon="el-icon-edit"
              @click="handleUpdate(scope.row)"
              v-hasPermi="['teaManager:teaManager:edit']"
            >修改</el-button>
            <el-button
              size="mini"
```

```
            type="text"
            icon="el-icon-delete"
            @click="handleDelete(scope.row)"
            v-hasPermi="['teaManager:teaManager:remove']"
          >删除</el-button>
        </template>
    </el-table-column>
</el-table>

<pagination
    v-show="total>0"
    :total="total"
    :page.sync="queryParams.pageNum"
    :limit.sync="queryParams.pageSize"
    @pagination="getList"
/>

<!-- 添加或修改教师对话框 -->
<el-dialog :title="title" :visible.sync="open" width="500px" append-to-body>
    <el-form ref="form" :model="form" :rules="rules" label-width="80px">
        <el-form-item label="教师所在学校 id" prop="schId">
            <el-input v-model="form.schId" placeholder="请输入教师所在学校 id" />
        </el-form-item>
        <el-form-item label="教师工号" prop="teaNum">
            <el-input v-model="form.teaNum" placeholder="请输入教师工号" />
        </el-form-item>
        <el-form-item label="教师名称" prop="teaName">
```

```
      <el-input v-model="form.teaName" placeholder="请输入教师名称" />
    </el-form-item>
    <el-form-item label="教师性别" prop="teaSex">
      <el-radio-group v-model="form.teaSex">
        <el-radio
          v-for="dict in dict.type.sys_user_sex"
          :key="dict.value"
          :label="dict.value"
        >{{dict.label}}</el-radio>
      </el-radio-group>
    </el-form-item>
    <el-form-item label="教师密码" prop="teaPwd">
      <el-input v-model="form.teaPwd" placeholder="请输入教师密码" />
    </el-form-item>
    <el-form-item label="教师电话" prop="teaPhone">
      <el-input v-model="form.teaPhone" placeholder="请输入教师电话" />
    </el-form-item>
    <el-form-item label="教师邮箱" prop="teaEmail">
      <el-input v-model="form.teaEmail" placeholder="请输入教师邮箱" />
    </el-form-item>
    <el-form-item label="教师头像" prop="teaAvatar">
      <image-upload v-model="form.teaAvatar"/>
    </el-form-item>
  </el-form>
  <div slot="footer" class="dialog-footer">
    <el-button type="primary" @click="submitForm">确 定</el-button>
    <el-button @click="cancel">取 消</el-button>
```

```
        </div>
      </el-dialog>
    </div>
</template>

<script>
import { listTeaManager, getTeaManager, delTeaManager, addTeaManager, updateTeaManager } from "@/api/teaManager/teaManager";

export default {
  name: "TeaManager",
  dicts: ['sys_user_sex'],
  data() {
    return {
      // 遮罩层
      loading: true,
      // 选中数组
      ids: [],
      // 非单个禁用
      single: true,
      // 非多个禁用
      multiple: true,
      // 显示搜索条件
      showSearch: true,
      // 总条数
      total: 0,
      // 教师表格数据
```

```
teaManagerList: [],
// 弹出层标题
title: "",
// 是否显示弹出层
open: false,
// 查询参数
queryParams: {
  pageNum: 1,
  pageSize: 10,
  teaNum: null,
  teaName: null,
  teaSex: null,
  teaPhone: null,
  teaEmail: null,
},
// 表单参数
form: {},
// 表单校验
rules: {
  teaId: [
    { required: true, message: "教师 id 不能为空", trigger: "blur" }
  ],
  schId: [
    { required: true, message: "教师所在学校 id 不能为空", trigger: "blur" }
  ],
  teaNum: [
    { required: true, message: "教师工号不能为空", trigger: "blur" }
```

```
        ],
        teaName: [
          { required: true, message: "教师名称不能为空", trigger: "blur" }
        ],
        teaSex: [
          { required: true, message: "教师性别不能为空", trigger: "change" }
        ],
        teaPwd: [
          { required: true, message: "教师密码不能为空", trigger: "blur" }
        ],
        teaAvatar: [
          { required: true, message: "教师头像不能为空", trigger: "blur" }
        ]
      }
    };
},
created() {
    this.getList();
},
methods: {
    /** 查询教师列表 */
    getList() {
        this.loading = true;
        listTeaManager(this.queryParams).then(response => {
            this.teaManagerList = response.rows;
            this.total = response.total;
            this.loading = false;
```

```
    });
},
// 取消按钮
cancel() {
    this.open = false;
    this.reset();
},
// 表单重置
reset() {
    this.form = {
        teaId: null,
        schId: null,
        teaNum: null,
        teaName: null,
        teaSex: null,
        teaPwd: null,
        teaPhone: null,
        teaEmail: null,
        teaAvatar: null
    };
    this.resetForm("form");
},
/** 搜索按钮操作 */
handleQuery() {
    this.queryParams.pageNum = 1;
    this.getList();
},
```

```
/** 重置按钮操作 */
resetQuery() {
    this.resetForm("queryForm");
    this.handleQuery();
},
// 多选框选中数据
handleSelectionChange(selection) {
    this.ids = selection.map(item => item.teaId)
    this.single = selection.length!==1
    this.multiple = !selection.length
},
/** 新增按钮操作 */
handleAdd() {
    this.reset();
    this.open = true;
    this.title = "添加教师";
},
/** 修改按钮操作 */
handleUpdate(row) {
    this.reset();
    const teaId = row.teaId || this.ids
    getTeaManager(teaId).then(response => {
        this.form = response.data;
        this.open = true;
        this.title = "修改教师";
    });
},
```

```
    /** 提交按钮 */
    submitForm() {
      this.$refs["form"].validate(valid => {
        if (valid) {
          if (this.form.teaId != null) {
            updateTeaManager(this.form).then(response => {
              this.$modal.msgSuccess("修改成功");
              this.open = false;
              this.getList();
            });
          } else {
            addTeaManager(this.form).then(response => {
              this.$modal.msgSuccess("新增成功");
              this.open = false;
              this.getList();
            });
          }
        }
      });
    },
    /** 删除按钮操作 */
    handleDelete(row) {
      const teaIds = row.teaId || this.ids;
      this.$modal.confirm('是否确认删除教师编号为"' + teaIds + '"的数据项?
').then(function() {
        return delTeaManager(teaIds);
      }).then(() => {
```

```
            this.getList();
            this.$modal.msgSuccess("删除成功");
          }).catch(() => {});
      },
      /** 导出按钮操作 */
      handleExport() {
        this.download('teaManager/teaManager/export', {
          ...this.queryParams
        }, `teaManager_${new Date().getTime()}.xlsx`)
      }
    }
};
</script>
```

src/view/dashboard/BarChart.vue

```
<template>
  <div :class="className" :style="{height:height,width:width}" />
</template>

<script>
import * as echarts from 'echarts'
require('echarts/theme/macarons') // echarts theme
import resize from './mixins/resize'

const animationDuration = 6000

export default {
```

```
mixins: [resize],
props: {
    className: {
        type: String,
        default: 'chart'
    },
    width: {
        type: String,
        default: '100%'
    },
    height: {
        type: String,
        default: '300px'
    }
},
data() {
    return {
        chart: null
    }
},
mounted() {
    this.$nextTick(() => {
        this.initChart()
    })
},
beforeDestroy() {
    if (!this.chart) {
```

```
        return
    }
    this.chart.dispose()
    this.chart = null
},
methods: {
    initChart() {
        this.chart = echarts.init(this.$el, 'macarons')

        this.chart.setOption({
            tooltip: {
                trigger: 'axis',
                axisPointer: { // 坐标轴指示器，坐标轴触发有效
                    type: 'shadow' // 默认为直线，可选为：'line' | 'shadow'
                }
            },
            grid: {
                top: 10,
                left: '2%',
                right: '2%',
                bottom: '3%',
                containLabel: true
            },
            xAxis: [{
                type: 'category',
                data: ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'],
                axisTick: {
```

```
        alignWithLabel: true
    }
}],
yAxis: [{
    type: 'value',
    axisTick: {
        show: false
    }
}],
series: [{
    name: 'pageA',
    type: 'bar',
    stack: 'vistors',
    barWidth: '60%',
    data: [79, 52, 200, 334, 390, 330, 220],
    animationDuration
}, {
    name: 'pageB',
    type: 'bar',
    stack: 'vistors',
    barWidth: '60%',
    data: [80, 52, 200, 334, 390, 330, 220],
    animationDuration
}, {
    name: 'pageC',
    type: 'bar',
    stack: 'vistors',
```

```
                barWidth: '60%',
                data: [30, 52, 200, 334, 390, 330, 220],
                animationDuration
            }]
        })
    }
  }
}
</script>
```

src/view/dashboard/LineChart.vue

```
<template>
  <div :class="className" :style="{height:height,width:width}" />
</template>

<script>
import * as echarts from 'echarts'
require('echarts/theme/macarons') // echarts theme
import resize from './mixins/resize'

export default {
  mixins: [resize],
  props: {
    className: {
      type: String,
      default: 'chart'
```

```
    },
    width: {
        type: String,
        default: '100%'
    },
    height: {
        type: String,
        default: '350px'
    },
    autoResize: {
        type: Boolean,
        default: true
    },
    chartData: {
        type: Object,
        required: true
    }
},
data() {
    return {
        chart: null
    }
},
watch: {
    chartData: {
        deep: true,
        handler(val) {
```

```
            this.setOptions(val)
        }
    }
},
mounted() {
    this.$nextTick(() => {
        this.initChart()
    })
},
beforeDestroy() {
    if (!this.chart) {
        return
    }
    this.chart.dispose()
    this.chart = null
},
methods: {
    initChart() {
        this.chart = echarts.init(this.$el, 'macarons')
        this.setOptions(this.chartData)
    },
    setOptions({ expectedData, actualData } = {}) {
        this.chart.setOption({
            xAxis: {
                data: ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'],
                boundaryGap: false,
                axisTick: {
```

```
        show: false
    }
},
grid: {
    left: 10,
    right: 10,
    bottom: 20,
    top: 30,
    containLabel: true
},
tooltip: {
    trigger: 'axis',
    axisPointer: {
        type: 'cross'
    },
    padding: [5, 10]
},
yAxis: {
    axisTick: {
        show: false
    }
},
legend: {
    data: ['expected', 'actual']
},
series: [{
    name: 'expected', itemStyle: {
```

```
        normal: {
            color: '#FF005A',
            lineStyle: {
                color: '#FF005A',
                width: 2
            }
        }
    },
    smooth: true,
    type: 'line',
    data: expectedData,
    animationDuration: 2800,
    animationEasing: 'cubicInOut'
},
{
    name: 'actual',
    smooth: true,
    type: 'line',
    itemStyle: {
        normal: {
            color: '#3888fa',
            lineStyle: {
                color: '#3888fa',
                width: 2
            },
            areaStyle: {
                color: '#f3f8ff'
```

```
                }
              }
            },
              data: actualData,
              animationDuration: 2800,
              animationEasing: 'quadraticOut'
          }]
        })
      }
    }
}
</script>
```

src/view/dashboard/PanelGroup.vue

```
<template>
  <el-row :gutter="40" class="panel-group">
    <el-col :xs="12" :sm="12" :lg="6" class="card-panel-col">
      <div class="card-panel" @click="handleSetLineChartData('newVisitis')">
        <div class="card-panel-icon-wrapper icon-people">
          <svg-icon icon-class="peoples" class-name="card-panel-icon" />
        </div>
        <div class="card-panel-description">
          <div class="card-panel-text">
            访客
          </div>
          <count-to :start-val="0" :end-val="102400" :duration="2600" class="card-
```

```
panel-num" />
            </div>
          </div>
        </el-col>
        <el-col :xs="12" :sm="12" :lg="6" class="card-panel-col">
          <div class="card-panel" @click="handleSetLineChartData('messages')">
            <div class="card-panel-icon-wrapper icon-message">
              <svg-icon icon-class="message" class-name="card-panel-icon" />
            </div>
            <div class="card-panel-description">
              <div class="card-panel-text">
                消息
              </div>
              <count-to :start-val="0" :end-val="81212" :duration="3000" class="card-
panel-num" />
            </div>
          </div>
        </el-col>
        <el-col :xs="12" :sm="12" :lg="6" class="card-panel-col">
          <div class="card-panel" @click="handleSetLineChartData('purchases')">
            <div class="card-panel-icon-wrapper icon-money">
              <svg-icon icon-class="money" class-name="card-panel-icon" />
            </div>
            <div class="card-panel-description">
              <div class="card-panel-text">
                金额
              </div>
```

```
            <count-to :start-val="0" :end-val="9280" :duration="3200" class="card-panel-num" />
          </div>
        </div>
      </el-col>
      <el-col :xs="12" :sm="12" :lg="6" class="card-panel-col">
        <div class="card-panel" @click="handleSetLineChartData('shoppings')">
          <div class="card-panel-icon-wrapper icon-shopping">
            <svg-icon icon-class="shopping" class-name="card-panel-icon" />
          </div>
          <div class="card-panel-description">
            <div class="card-panel-text">
              订单
            </div>
            <count-to :start-val="0" :end-val="13600" :duration="3600" class="card-panel-num" />
          </div>
        </div>
      </el-col>
    </el-row>
</template>

<script>
import CountTo from 'vue-count-to'

export default {
  components: {
```

```
        CountTo
    },
    methods: {
        handleSetLineChartData(type) {
            this.$emit('handleSetLineChartData', type)
        }
    }
}
</script>

<style lang="scss" scoped>
.panel-group {
    margin-top: 18px;

    .card-panel-col {
        margin-bottom: 32px;
    }

    .card-panel {
        height: 108px;
        cursor: pointer;
        font-size: 12px;
        position: relative;
        overflow: hidden;
        color: #666;
        background: #fff;
        box-shadow: 4px 4px 40px rgba(0, 0, 0, .05);
```

```
border-color: rgba(0, 0, 0, .05);

&:hover {
    .card-panel-icon-wrapper {
        color: #fff;
    }

    .icon-people {
        background: #40c9c6;
    }

    .icon-message {
        background: #36a3f7;
    }

    .icon-money {
        background: #f4516c;
    }

    .icon-shopping {
        background: #34bfa3
    }
}

.icon-people {
    color: #40c9c6;
}
```

```
.icon-message {
    color: #36a3f7;
}

.icon-money {
    color: #f4516c;
}

.icon-shopping {
    color: #34bfa3
}

.card-panel-icon-wrapper {
    float: left;
    margin: 14px 0 0 14px;
    padding: 16px;
    transition: all 0.38s ease-out;
    border-radius: 6px;
}

.card-panel-icon {
    float: left;
    font-size: 48px;
}

.card-panel-description {
```

```
            float: right;
            font-weight: bold;
            margin: 26px;
            margin-left: 0px;

            .card-panel-text {
                line-height: 18px;
                color: rgba(0, 0, 0, 0.45);
                font-size: 16px;
                margin-bottom: 12px;
            }

            .card-panel-num {
                font-size: 20px;
            }
        }
    }
}

@media (max-width:550px) {
    .card-panel-description {
        display: none;
    }

    .card-panel-icon-wrapper {
        float: none !important;
        width: 100%;
```

```
        height: 100%;
        margin: 0 !important;

        .svg-icon {
            display: block;
            margin: 14px auto !important;
            float: none !important;
        }
    }
}
</style>
```

src/view/dashboard/PieChart.vue

```
<template>
    <div :class="className" :style="{height:height,width:width}" />
</template>

<script>
import * as echarts from 'echarts'
require('echarts/theme/macarons') // echarts theme
import resize from './mixins/resize'

export default {
    mixins: [resize],
    props: {
        className: {
```

```
        type: String,
        default: 'chart'
    },
    width: {
        type: String,
        default: '100%'
    },
    height: {
        type: String,
        default: '300px'
    }
},
data() {
    return {
        chart: null
    }
},
mounted() {
    this.$nextTick(() => {
        this.initChart()
    })
},
beforeDestroy() {
    if (!this.chart) {
        return
    }
    this.chart.dispose()
```

```
    this.chart = null
},
methods: {
    initChart() {
        this.chart = echarts.init(this.$el, 'macarons')

        this.chart.setOption({
            tooltip: {
                trigger: 'item',
                formatter: '{a} <br/>{b} : {c} ({d}%)'
            },
            legend: {
                left: 'center',
                bottom: '10',
                data: ['Industries', 'Technology', 'Forex', 'Gold', 'Forecasts']
            },
            series: [
                {
                    name: 'WEEKLY WRITE ARTICLES',
                    type: 'pie',
                    roseType: 'radius',
                    radius: [15, 95],
                    center: ['50%', '38%'],
                    data: [
                        { value: 320, name: 'Industries' },
                        { value: 240, name: 'Technology' },
                        { value: 149, name: 'Forex' },
```

```
                { value: 100, name: 'Gold' },
                { value: 59, name: 'Forecasts' }
              ],
              animationEasing: 'cubicInOut',
              animationDuration: 2600
            }
          ]
        })
      }
    }
}
</script>
```

src/view/dashboard/RaddarChart.vue

```
<template>
  <div :class="className" :style="{height:height,width:width}" />
</template>

<script>
import * as echarts from 'echarts'
require('echarts/theme/macarons') // echarts theme
import resize from './mixins/resize'

const animationDuration = 3000

export default {
```

```
mixins: [resize],
props: {
    className: {
        type: String,
        default: 'chart'
    },
    width: {
        type: String,
        default: '100%'
    },
    height: {
        type: String,
        default: '300px'
    }
},
data() {
    return {
        chart: null
    }
},
mounted() {
    this.$nextTick(() => {
        this.initChart()
    })
},
beforeDestroy() {
    if (!this.chart) {
```

```
        return
    }
    this.chart.dispose()
    this.chart = null
},
methods: {
    initChart() {
        this.chart = echarts.init(this.$el, 'macarons')

        this.chart.setOption({
            tooltip: {
                trigger: 'axis',
                axisPointer: { // 坐标轴指示器，坐标轴触发有效
                    type: 'shadow' // 默认为直线，可选为：'line' | 'shadow'
                }
            },
            radar: {
                radius: '66%',
                center: ['50%', '42%'],
                splitNumber: 8,
                splitArea: {
                    areaStyle: {
                        color: 'rgba(127,95,132,.3)',
                        opacity: 1,
                        shadowBlur: 45,
                        shadowColor: 'rgba(0,0,0,.5)',
                        shadowOffsetX: 0,
```

```
            shadowOffsetY: 15
        }
    },
    indicator: [
        { name: 'Sales', max: 10000 },
        { name: 'Administration', max: 20000 },
        { name: 'Information Techology', max: 20000 },
        { name: 'Customer Support', max: 20000 },
        { name: 'Development', max: 20000 },
        { name: 'Marketing', max: 20000 }
    ]
},
legend: {
    left: 'center',
    bottom: '10',
    data: ['Allocated Budget', 'Expected Spending', 'Actual Spending']
},
series: [{
    type: 'radar',
    symbolSize: 0,
    areaStyle: {
        normal: {
            shadowBlur: 13,
            shadowColor: 'rgba(0,0,0,.2)',
            shadowOffsetX: 0,
            shadowOffsetY: 10,
            opacity: 1
```

```
            }
          },
          data: [
            {
              value: [5000, 7000, 12000, 11000, 15000, 14000],
              name: 'Allocated Budget'
            },
            {
              value: [4000, 9000, 15000, 15000, 13000, 11000],
              name: 'Expected Spending'
            },
            {
              value: [5500, 11000, 12000, 15000, 12000, 12000],
              name: 'Actual Spending'
            }
          ],
          animationDuration: animationDuration
        }]
      })
    }
  }
}
</script>
```

api/login.js

```
import request from '@/utils/request'
```

```
// 登录方法
export function login(username, password, code, uuid) {
  const data = {
    username,
    password,
    code,
    uuid
  }
  return request({
    url: '/login',
    headers: {
      isToken: false,
      repeatSubmit: false
    },
    method: 'post',
    data: data
  })
}

// 注册方法
export function register(data) {
  return request({
    url: '/register',
    headers: {
      isToken: false
    },
    method: 'post',
```

```
        data: data
    })
}

// 获取用户详细信息
export function getInfo() {
    return request({
        url: '/getInfo',
        method: 'get'
    })
}

// 退出方法
export function logout() {
    return request({
        url: '/logout',
        method: 'post'
    })
}

// 获取验证码
export function getCodeImg() {
    return request({
        url: '/captchaImage',
        headers: {
            isToken: false
        },
```

```
        method: 'get',
        timeout: 20000
    })
}
```

api/menu.js

```
import request from '@/utils/request'

// 获取路由
export const getRouters = () => {
    return request({
        url: '/getRouters',
        method: 'get'
    })
}
```

api/courseManager/course.js

```
import request from '@/utils/request'

// 查询课程管理列表
export function listCourse(query) {
    return request({
        url: '/courseManager/course/list',
        method: 'get',
        params: query
    })
}
```

```
// 查询课程管理详细
export function getCourse(courseId) {
  return request({
    url: '/courseManager/course/' + courseId,
    method: 'get'
  })
}

// 新增课程管理
export function addCourse(data) {
  return request({
    url: '/courseManager/course',
    method: 'post',
    data: data
  })
}

// 修改课程管理
export function updateCourse(data) {
  return request({
    url: '/courseManager/course',
    method: 'put',
    data: data
  })
}
```

```
// 删除课程管理
export function delCourse(courseId) {
  return request({
    url: '/courseManager/course/' + courseId,
    method: 'delete'
  })
}
```

api/machineManager/machine.js

```
import request from '@/utils/request'

// 查询机器管理列表
export function listMachine(query) {
  return request({
    url: '/machineManager/machine/list',
    method: 'get',
    params: query
  })
}

// 查询机器管理详细
export function getMachine(macId) {
  return request({
    url: '/machineManager/machine/' + macId,
    method: 'get'
  })
```

```
}

// 新增机器管理
export function addMachine(data) {
    return request({
        url: '/machineManager/machine',
        method: 'post',
        data: data
    })
}

// 修改机器管理
export function updateMachine(data) {
    return request({
        url: '/machineManager/machine',
        method: 'put',
        data: data
    })
}

// 删除机器管理
export function delMachine(macId) {
    return request({
        url: '/machineManager/machine/' + macId,
        method: 'delete'
    })
}
```

api/phytestManager/phytest.js

```
import request from '@/utils/request'

// 查询体测项目管理列表
export function listPhytest(query) {
  return request({
    url: '/phytestManager/phytest/list',
    method: 'get',
    params: query
  })
}

// 查询体测项目管理详细
export function getPhytest(ptId) {
  return request({
    url: '/phytestManager/phytest/' + ptId,
    method: 'get'
  })
}

// 新增体测项目管理
export function addPhytest(data) {
  return request({
    url: '/phytestManager/phytest',
    method: 'post',
```

```
        data: data
    })
}

// 修改体测项目管理
export function updatePhytest(data) {
    return request({
        url: '/phytestManager/phytest',
        method: 'put',
        data: data
    })
}

// 删除体测项目管理
export function delPhytest(ptId) {
    return request({
        url: '/phytestManager/phytest/' + ptId,
        method: 'delete'
    })
}
```

api/ptbookManager/ptbook.js

```
import request from '@/utils/request'

// 查询体测预约管理列表
export function listPtbook(query) {
```

```
    return request({
        url: '/ptbookManager/ptbook/list',
        method: 'get',
        params: query
    })
}

// 查询体测预约管理详细
export function getPtbook(ptbookId) {
    return request({
        url: '/ptbookManager/ptbook/' + ptbookId,
        method: 'get'
    })
}

// 新增体测预约管理
export function addPtbook(data) {
    return request({
        url: '/ptbookManager/ptbook',
        method: 'post',
        data: data
    })
}

// 修改体测预约管理
export function updatePtbook(data) {
    return request({
```

```
    url: '/ptbookManager/ptbook',
    method: 'put',
    data: data
  })
}

// 删除体测预约管理
export function delPtbook(ptbookId) {
  return request({
    url: '/ptbookManager/ptbook/' + ptbookId,
    method: 'delete'
  })
}
```

api/ptrecordManager/ptrecord.js

```
import request from '@/utils/request'

// 查询体测记录管理列表
export function listPtrecord(query) {
  return request({
    url: '/ptrecordManager/ptrecord/list',
    method: 'get',
    params: query
  })
}
```

```
// 查询体测记录管理详细
export function getPtrecord(ptrId) {
  return request({
    url: '/ptrecordManager/ptrecord/' + ptrId,
    method: 'get'
  })
}

// 新增体测记录管理
export function addPtrecord(data) {
  return request({
    url: '/ptrecordManager/ptrecord',
    method: 'post',
    data: data
  })
}

// 修改体测记录管理
export function updatePtrecord(data) {
  return request({
    url: '/ptrecordManager/ptrecord',
    method: 'put',
    data: data
  })
}

// 删除体测记录管理
```

```
export function delPtrecord(ptrId) {
    return request({
        url: '/ptrecordManager/ptrecord/' + ptrId,
        method: 'delete'
    })
}
```

api/ptscoreManager/ptscore.js

```
import request from '@/utils/request'

// 查询学生分数管理列表
export function listPtscore(query) {
    return request({
        url: '/ptscoreManager/ptscore/list',
        method: 'get',
        params: query
    })
}

// 查询学生分数管理详细
export function getPtscore(ptscoreId) {
    return request({
        url: '/ptscoreManager/ptscore/' + ptscoreId,
        method: 'get'
    })
}
```

```
// 新增学生分数管理
export function addPtscore(data) {
  return request({
    url: '/ptscoreManager/ptscore',
    method: 'post',
    data: data
  })
}

// 修改学生分数管理
export function updatePtscore(data) {
  return request({
    url: '/ptscoreManager/ptscore',
    method: 'put',
    data: data
  })
}

// 删除学生分数管理
export function delPtscore(ptscoreId) {
  return request({
    url: '/ptscoreManager/ptscore/' + ptscoreId,
    method: 'delete'
  })
}
```

api/ptstandManager/ptstandManager.js

```
import request from '@/utils/request'

// 查询体测标准列表
export function listPtstandManager(query) {
  return request({
    url: '/ptstandManager/ptstandManager/list',
    method: 'get',
    params: query
  })
}

// 查询体测标准详细
export function getPtstandManager(standId) {
  return request({
    url: '/ptstandManager/ptstandManager/' + standId,
    method: 'get'
  })
}

// 新增体测标准
export function addPtstandManager(data) {
  return request({
    url: '/ptstandManager/ptstandManager',
    method: 'post',
    data: data
```

```
  })
}

// 修改体测标准
export function updatePtstandManager(data) {
  return request({
    url: '/ptstandManager/ptstandManager',
    method: 'put',
    data: data
  })
}

// 删除体测标准
export function delPtstandManager(standId) {
  return request({
    url: '/ptstandManager/ptstandManager/' + standId,
    method: 'delete'
  })
}
```

api/stuManager/stu.js

```
import request from '@/utils/request'

// 查询学生信息管理列表
export function listStu(query) {
  return request({
```

```
        url: '/stuManager/stu/list',
        method: 'get',
        params: query
    })
}

// 查询学生信息管理详细
export function getStu(stuId) {
    return request({
        url: '/stuManager/stu/' + stuId,
        method: 'get'
    })
}

// 新增学生信息管理
export function addStu(data) {
    return request({
        url: '/stuManager/stu',
        method: 'post',
        data: data
    })
}

// 修改学生信息管理
export function updateStu(data) {
    return request({
        url: '/stuManager/stu',
```

```
        method: 'put',
        data: data
    })
}

// 删除学生信息管理
export function delStu(stuId) {
    return request({
        url: '/stuManager/stu/' + stuId,
        method: 'delete'
    })
}
```

api/teaManager/teaManager.js

```
import request from '@/utils/request'

// 查询教师列表
export function listTeaManager(query) {
    return request({
        url: '/teaManager/teaManager/list',
        method: 'get',
        params: query
    })
}

// 查询教师详细
```

```
export function getTeaManager(teaId) {
    return request({
        url: '/teaManager/teaManager/' + teaId,
        method: 'get'
    })
}

// 新增教师
export function addTeaManager(data) {
    return request({
        url: '/teaManager/teaManager',
        method: 'post',
        data: data
    })
}

// 修改教师
export function updateTeaManager(data) {
    return request({
        url: '/teaManager/teaManager',
        method: 'put',
        data: data
    })
}

// 删除教师
export function delTeaManager(teaId) {
```

```
    return request({
        url: '/teaManager/teaManager/' + teaId,
        method: 'delete'
    })
}
```