项目实训总结报告

项 目 名 称: TFG医院管理系统

学 校 名 称: 河南牧业经济学院

项 目 讲 师: 贾志贤

姓 名: 曹帅奇

学 号: 2102160601002

实 训 日 期: 2023年6月26日-7月7日

**1 引言**

1.1项目概述

项目名称：医院管理系统 - 科室管理模块

项目目的： 该项目旨在实现医院管理系统中的科室管理功能，提供对医院科室信息的增删改查操作。通过该模块，医院管理人员可以方便地管理和维护科室信息，包括科室编码、科室名称等。

背景： 在传统的医院管理中，科室管理是一个重要的环节。医院需要对科室进行有效的管理和组织，以保证医疗服务的顺利运行。传统的科室管理方式可能面临信息不便捷、效率低下等问题。因此，开发一个科室管理模块，利用信息化技术提高科室管理效率和精确度，具有重要的意义。

意义： 通过该科室管理模块，医院管理人员可以方便地进行科室信息的增删改查操作，提高科室管理的效率和准确性。同时，该模块还可以提供科室信息的统计和分析功能，帮助医院管理人员更好地了解科室情况，为决策提供依据。科室管理模块的实施可以提高医院管理水平，优化医疗资源的配置，提升医院整体运营效益。

1.2 需求分析

为什么做这个题目：

* 提高医院科室管理的效率：传统的科室管理方式可能需要手动记录和维护科室信息，效率较低。通过开发科室管理模块，可以利用信息化技术实现科室信息的自动化管理，提高管理效率。
* 精确掌握科室情况：科室管理模块可以提供科室信息的统计和分析功能，帮助医院管理人员了解科室的运营情况，为决策提供依据。
* 优化医院资源配置：科室管理模块可以帮助医院管理人员合理配置医疗资源，提高医疗服务的质量和效果。

实际价值：

* 提高医院管理效率：科室管理模块的实施可以简化和自动化科室管理流程，减少人力成本和时间消耗，提高管理效率。
* 提升医院服务质量：科室管理模块可以帮助医院更好地组织和管理科室，确保医疗服务的顺畅进行，提升医院的服务质量和患者满意度。
* 数据统计和分析：科室管理模块可以提供科室信息的统计和分析功能，帮助医院管理人员深入了解科室的运营情况和业务特点，为决策提供数据支持。
* 资源优化和配置：科室管理模块可以帮助医院管理人员优化医疗资源的配置，合理分配人力、设备和物资，提高资源利用率和医疗效益。
* 系统集成和扩展：科室管理模块作为医院管理系统的一部分，可以与其他模块进行集成，实现信息共享和协同工作，为医院整体管理提供支持。

通过科室管理模块的开发和实施，医院可以更加高效地管理科室信息，提升医疗服务质量，优化资源配置，进一步提高医院的竞争力和运营效益。

1.3 运行环境

**软件环境：**

**- 操作系统：Windows11**

**- Java运行环境：JDK 8**

**- 开发工具：IntelliJ IDEA，Visual Studio Code**

**- 数据库：MySQL**

**- 项目管理工具：Maven**

**2 项目设计**

2.1 设计思路

设计思路主要包括以下几个方面：

* 数据库设计：设计科室信息的数据表结构，包括科室ID、科室编码、科室名称等字段，以满足科室管理的需求。
* 后端架构设计：采用Spring Boot框架搭建后端服务，使用MyBatis-Plus作为持久层框架，实现科室信息的增删改查操作。
* 前端界面设计：使用前端框架（如Vue.js）搭建科室管理模块的用户界面，实现用户对科室信息的可视化操作和展示。

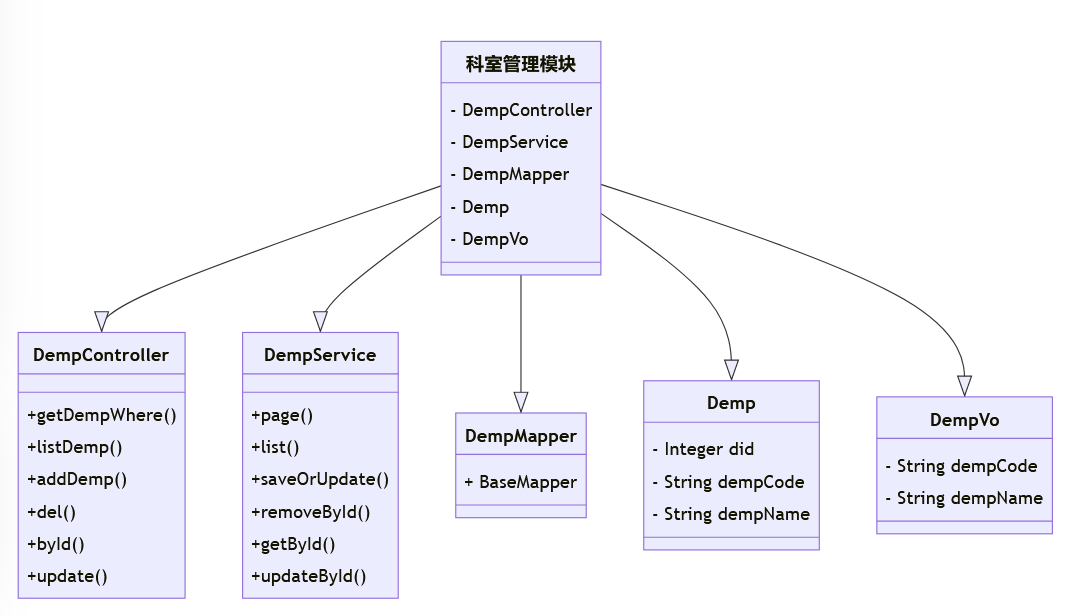
2.2 模块功能介绍

模块功能介绍是对科室管理模块的功能进行详细说明，包括模块提供的主要功能和操作流程。具体的功能介绍可以根据项目需求进行补充。

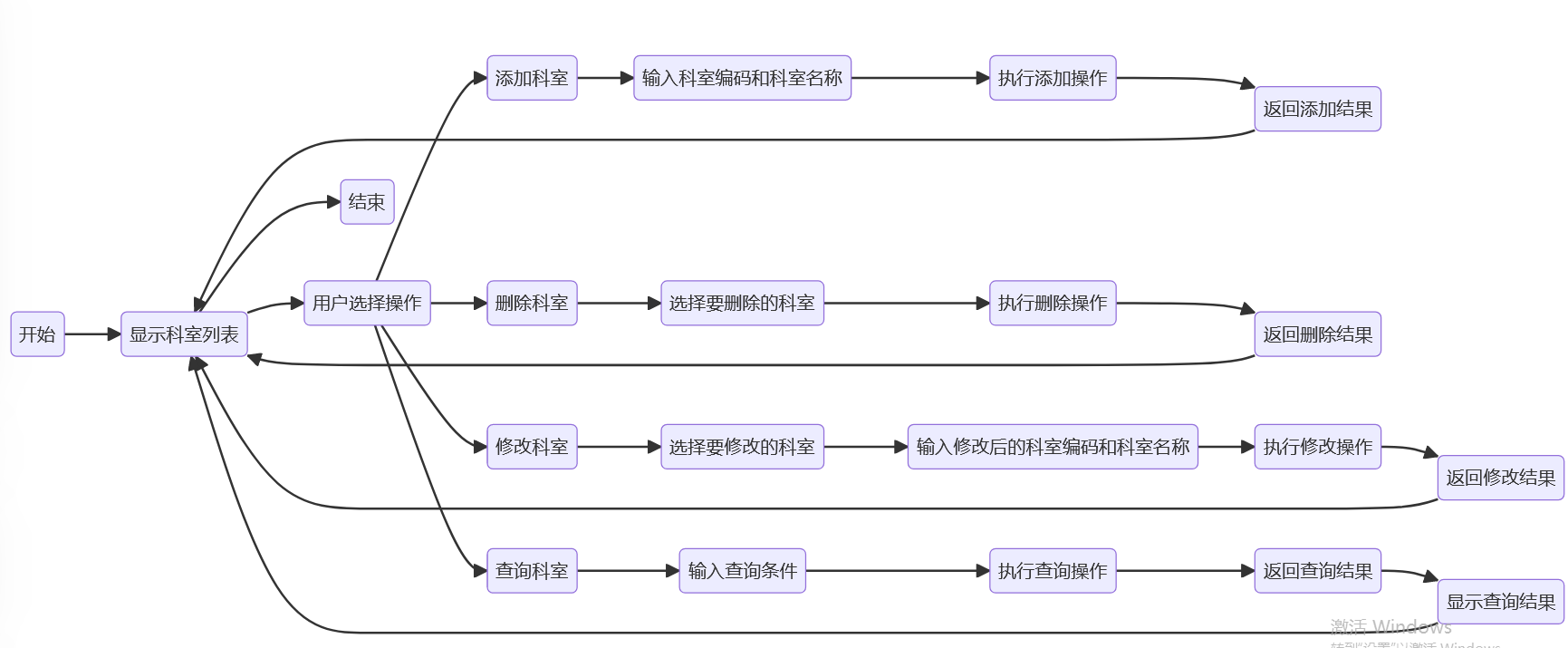
科室管理模块的主要功能包括：

* 添加科室：允许用户添加新的科室信息，包括科室编码和科室名称。
* 删除科室：允许用户根据科室ID删除指定的科室信息。
* 修改科室：允许用户修改科室的编码和名称。
* 查询科室：允许用户根据条件（如科室编码、科室名称）进行科室信息的查询，支持分页显示查询结果。

2.3 模块结构图



2.4 程序流程图



2.5 功能设计分工

在科室管理模块的功能设计分工中，我负责全部设计和开发工作

1. 控制器（DempController）：作为模块的入口，负责接收用户请求并进行相应的业务逻辑处理。您将独自负责该控制器的设计和实现，包括处理添加、删除、修改、查询等操作的请求，以及返回相应的结果。

2. 服务（DempService）：作为业务逻辑的实现层，负责具体的数据操作和相关的业务处理。您将独自负责该服务类的设计和实现，包括调用数据访问层（DempMapper）进行数据操作、处理业务逻辑、返回结果等。

3. 数据访问层（DempMapper）：负责与数据库进行交互，执行增删改查操作。您将独自负责该Mapper接口的设计和实现，包括编写SQL语句、使用MyBatis-Plus进行数据操作等。

4. 实体类（Demp）和查询参数（DempVo）：负责定义数据模型和查询参数模型。您将独自负责这两个类的设计和实现，包括定义字段、添加注解（如@TableId、@TableName等）、编写属性的Getter和Setter方法等。

**3 详细设计**

3.1 系统设计

在科室管理模块的系统设计中，主要包括以下方面的设计内容：

1. 系统架构：采用Spring Boot框架进行开发，使用MVC（Model-View-Controller）架构模式进行系统设计，实现前后端分离。

2. 数据库设计：设计科室表（demp）存储科室的信息，包括科室ID（did）、科室编码（dempCode）、科室名称（dempName）等字段。

3. 接口设计：设计RESTful风格的接口，包括添加科室、删除科室、修改科室、查询科室等接口，通过HTTP方法（POST、DELETE、PUT、GET）进行操作。

4. 用户权限控制：使用Shiro框架实现用户的认证和授权功能，对科室管理操作进行权限控制。

5. 异常处理：设计全局异常处理器，统一处理系统中的异常情况，返回合适的错误码和错误信息。

3.2 数据库设计

在数据库设计中，设计了一个名为"demp"的表，用于存储科室的信息。表的字段设计如下：

- did：科室ID，使用自增长的整型作为主键。

- dempCode：科室编码，使用字符串类型存储科室的编码信息。

- dempName：科室名称，使用字符串类型存储科室的名称信息。

通过这样的设计，可以在数据库中存储和管理科室的基本信息。

3.3 核心技术功能描述

科室管理模块涉及的技术和核心功能描述如下：

- 技术：

- Spring Boot：使用Spring Boot框架进行快速开发和构建微服务。

- MyBatis-Plus：使用MyBatis-Plus简化对数据库的操作，提供了便捷的CRUD方法。

- MySQL：使用MySQL数据库存储和管理科室信息。

- Shiro：使用Shiro框架实现用户认证和授权，实现对科室管理操作的权限控制。

- RESTful API：设计和实现基于RESTful风格的接口，实现对科室的增删改查操作。

- 核心功能：

- 添加科室：实现向系统中添加新的科室信息。

- 删除科室：实现从系统中删除指定的科室信息。

- 修改科室：实现对已存在的科室信息进行修改。

- 查询科室：实现根据条件查询科室信息，并返回查询结果。

**4 实现方案**

本项目的实现方案是基于Spring Boot框架和MyBatis-Plus技术进行开发。以下是对各个功能的实现描述、核心代码以及实现结果的详细说明。

1. 实现科室列表查询功能：

- 代码部分：

```java

@GetMapping("/list")

public Result listDemp(){

List<Demp> dempList = dempService.list(null);

if(dempList == null){

return Result.fail().code(201).message("未查到数据");

}

return Result.ok(dempList);

}

```

- 实现描述：该方法使用`dempService`调用`list`方法查询所有科室信息，并将结果封装为`Result`对象返回。

- 核心代码解释：

- 通过`dempService.list(null)`调用`DempService`的`list`方法查询所有科室信息。

- 如果查询结果为null，返回一个失败的`Result`对象，包含错误码201和错误信息"未查到数据"。

- 如果查询结果不为null，返回一个成功的`Result`对象，包含科室列表数据。

- 实现结果：成功查询并返回科室列表数据。

2. 实现添加科室功能：

- 代码部分：

```java

@PostMapping("/addOrUpdate")

public Result addDemp(@RequestBody Demp demp){

boolean save = dempService.saveOrUpdate(demp);

if(save){

return Result.ok(save);

}

return Result.fail().code(201).message("添加失败");

}

```

- 实现描述：该方法接收一个`Demp`对象作为参数，调用`dempService`的`saveOrUpdate`方法进行科室添加操作，并根据操作结果返回相应的`Result`对象。

- 核心代码解释：

- 通过`dempService.saveOrUpdate(demp)`调用`DempService`的`saveOrUpdate`方法进行科室添加操作。

- 如果添加成功，返回一个成功的`Result`对象，包含操作结果。

- 如果添加失败，返回一个失败的`Result`对象，包含错误码201和错误信息"添加失败"。

- 实现结果：成功添加科室并返回操作结果。

3. 实现删除科室功能：

- 代码部分：

```java

@DeleteMapping("/del/{did}")

public Result del(@PathVariable Integer did){

boolean remove = dempService.removeById(did);

if(remove){

return Result.ok(remove);

}

return Result.fail().code(201).message("删除失败");

}

```

- 实现描述：该方法接收一个科室ID作为参数，调用`dempService`的`removeById`方法进行科室删除操作，并根据操作结果返回相应的`Result`对象。

- 核心代码解释：

- 通过`dempService.removeById(did)`调用`DempService`的`removeById`方法进行科室删除操作。

- 如果删除成功，返回一个成功的`Result`对象，包含操作结果。

- 如果删除失败，返回一个失败的`Result`对象，包含错误

每一步操作截图及说明：



1. 执行程序弹出界面：



说明：展示了科室管理系统的初始界面，用户可以在此界面进行科室管理操作。

3. 添加科室：



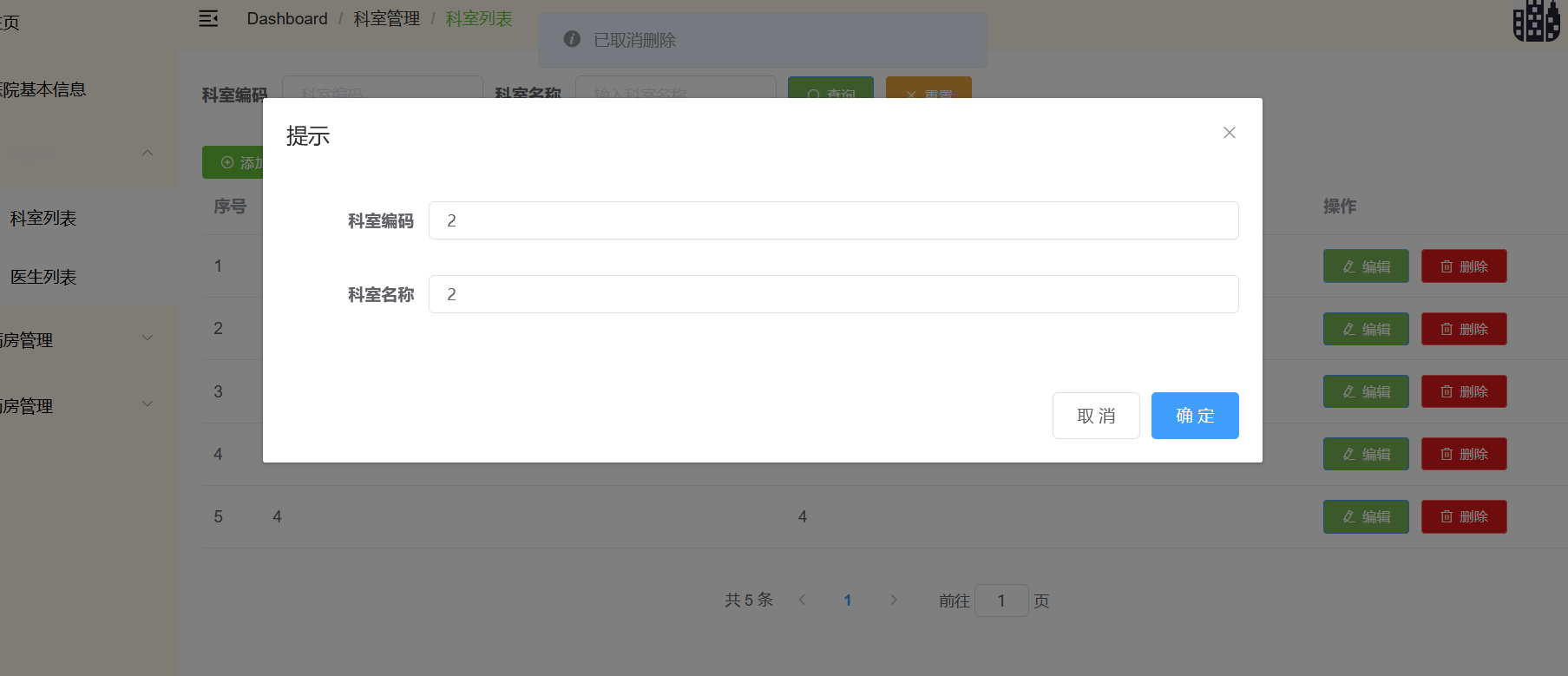
说明：用户点击"添加科室"按钮后，在弹出的对话框中输入科室编码和科室名称，点击"确认"按钮进行科室添加操作。

4. 删除科室：



说明：用户选择要删除的科室，并点击"删除科室"按钮进行科室删除操作。

5. 修改科室：

 说明：用户选择要修改的科室，并在弹出的对话框中输入修改后的科室编码和科室名称，点击"确认"按钮进行科室修改操作。

6. 查询科室：



1. 实现科室列表查询功能：

- 根据提供的`DempController`中的`listDemp()`方法，调用`dempService`的`list`方法查询所有科室信息。

- 将查询结果封装到`Result`对象中，并返回给前端展示。

2. 实现添加科室功能：

- 根据提供的`DempController`中的`addDemp()`方法，接收前端传入的科室信息。

- 调用`dempService`的`saveOrUpdate`方法将科室信息保存或更新到数据库中。

- 根据操作结果，将相应的操作成功或失败信息封装到`Result`对象中，并返回给前端展示。

3. 实现删除科室功能：

- 根据提供的`DempController`中的`del()`方法，接收前端传入的科室ID。

- 调用`dempService`的`removeById`方法根据科室ID进行科室删除操作。

- 根据删除操作的结果，将相应的操作成功或失败信息封装到`Result`对象中，并返回给前端展示。

4. 实现修改科室功能：

- 根据提供的`DempController`中的`update()`方法，接收前端传入的修改后的科室信息。

- 调用`dempService`的`updateById`方法根据科室ID进行科室信息的更新。

- 根据更新操作的结果，将相应的操作成功或失败信息封装到`Result`对象中，并返回给前端展示。

5. 实现查询科室功能：

- 根据提供的`DempController`中的`getDempWhere()`方法，接收前端传入的查询条件。

- 根据查询条件构造`QueryWrapper`对象，并调用`dempService`的`page`方法进行条件查询。

- 将查询结果封装到`Result`对象中，并返回给前端展示。

**5 系统测试**

测试方法：

1. 单元测试：对每个功能模块进行独立测试，验证其功能是否按照设计要求正确实现。

2. 集成测试：将各个功能模块组合在一起，测试它们之间的交互和协作是否正常。

3. 冒烟测试：测试系统的基本功能是否能够正常运行，检查系统是否存在严重问题。

测试描述：

1. 单元测试：针对每个功能模块编写相应的测试用例，包括正常输入、边界值输入、异常输入等情况，验证功能是否按照预期工作。

2. 集成测试：将各个功能模块组合在一起进行测试，检查它们之间的接口和数据交互是否正常，是否能够正确协同工作。

3. 冒烟测试：对系统的主要功能进行简单测试，验证系统是否能够正常启动，界面是否正常显示，基本功能是否可用。

预期结果：

1. 单元测试：每个功能模块的测试用例都能够通过，功能按照预期工作。

2. 集成测试：各个功能模块之间的交互和协作正常，系统整体功能符合预期。

3. 冒烟测试：系统能够正常启动，界面正常显示，主要功能可用。

**6 心得体会**

在实训过程中，我对项目开发的各个阶段有了更深入的了解。通过需求分析、设计、实现和测试等环节的学习和实践，我对项目开发的流程和方法有了更清晰的认识。

在设计阶段，我学会了如何将需求转化为可执行的设计方案，包括模块设计、数据设计和界面设计等。我了解到良好的设计是项目成功的基础，它能够提高代码的可维护性和可扩展性。

在实现阶段，我通过编写代码将设计方案转化为实际可运行的程序。我学会了如何合理地组织代码结构，编写高质量的代码，并且通过调试和测试来验证代码的正确性和稳定性。

总的来说，通过这次实训，我不仅学到了理论知识，还获得了实际项目开发的经验。我深刻体会到了团队合作的重要性，通过与团队成员的沟通和协作，我们能够共同克服困难，解决问题，使项目顺利进行。