****

**专业综合课程设计**

**软件需求规格说明书**

|  |  |  |  |
| --- | --- | --- | --- |
| 组长姓名 | 徐子衿 | 学号 | 16202202 |
| 专业 | 软件工程 | 班级 | 162022 |
| 成员 | 16202201 朱俊莹 16202203 杨文慧 | | |
| 课题名称 | 项目协作开发平台 | | |
| 指导教师 | 张恒峰 | | |
| 开课学期 | 2018~ 2019 第一学期 | | |

目录

1. 系统分析

1.1 需求概述

1.2 用例分析与描述

1.2.1 用例分析

1.2.2 用例描述

2. 系统设计

2.1 多层结构设计

2.2 问题域设计

2.3 持久化设计

2.4 界面设计

3. 系统实现

3.1 表达层

3.2 业务层

3.3 持久化层

4. 系统测试

4.1 单元测试

4.2 功能测试

4.3 性能测试（可选）

5. 总结与体会

字体说明：中文宋体、英文Times New Roman

1级标题 四号加粗

2级标题 小四加粗

3级标题 五号加粗

正文 五号

图号：[章节号]-[序号] [说明]，小五，位于图的下方，例如：图 1-5 用例图

表号：[章节号]-[序号] [说明]，小五，位于表的上方，例如：表 1-5 宠物表

# 1 系统分析

## 1.1 需求概述

项目背景

项目协作开发平台系统是一个面向程序员组队与开发的系统，在以往的项目组建中，程序员只能通过人工线下组队，通知会议等都很不方便。本系统极大的便捷了程序员们的组队，团队负责人可以在该系统上创建、管理项目并安排组员开展会议，组员可以进行查询会议时间、发布文档、代码整合维护等操作。

需求说明

随着计算机网络和信息技术的发展和壮大，程序员协作组队的信息录入与管理系统化已成为必然。本系统面向全国程序员，系统的便捷性大大提高了程序员组建项目的兴趣，对培养程序员的团队合作精神都发挥着积极的作用。因此，一个全面的项目协作开发平台系统，可以对团队的组建合作，对项目的编辑、评审、维护等各项过程起到综合协调，积极推动构建项目的各项工作。所以，本系统在对社会影响绝对是积极正面、完全可行的。

本系统严格遵守国家法律、法规对项目设计相对的要求，遵纪守法，并按照项目设计的有关国家标准、规范进行，确保项目设计的合规性、可行性、适应性。本系统严格按要求进行项目设计的评审，把好评审关，及时发现、处理问题，确保设计质量，做好与项目实施的衔接和设计交底，本系统完全由开发人员依靠所学知识与技术编写、修改、测试而来，不存在侵权抄袭行为，且项目协作开发平台系统对社会影响积极向上，故在法律方面是可行的。

业务规则

Here is a list of business rules:

* An owner may not have multiple pets with the same case-insensitive name.
* …

## 1.2 用例分析与描述

### 1.2.1 用例分析

先分析系统有哪些角色，然后说明这些角色能够完成哪些操作，最后形成用例图

团队负责人的用例：...

组员的用例：...

管理员的用例：

* 增加程序员
* 修改程序员信息
* 查询程序员信息
* 删除程序员



图1.1 宠物医院系统用例图

### 1.3.2 用例描述

给出每个用例的说明与详细流程

|  |  |
| --- | --- |
| 用例编号 | UC01 |
| 用例名称 | 增加程序员 |
| 简要说明 | 管理员增加程序员 |
| 执行者 | 管理员 |
| 前置条件 | 管理员已成功登录 |
| 后置条件 | 管理员成功增加程序员 |
| 基本事件流 | 1.管理员进入管理员功能页面  2.管理员选择增加程序员功能  3.系统提示输入程序员信息  4.管理员输入信息  A1:信息格式不符  重新输入信息  5.系统提示录入成功  6.用例结束 |
| 扩展事件流 | A1:信息格式不符  重新输入信息 |

|  |  |
| --- | --- |
| 用例编号 | UC02 |
| 用例名称 | 修改程序员信息 |
| 简要说明 | 管理员修改程序员信息 |
| 执行者 | 管理员 |
| 前置条件 | 管理员已成功登录 |
| 后置条件 | 管理员成功修改程序员信息 |
| 基本事件流 | 1.管理员进入管理员功能页面  2.管理员选择修改程序员信息功能  3.系统提示选择程序员  4.管理员输入修改信息  A1:信息格式不符  重新输入信息  5.系统提示录入成功  6.用例结束 |
| 扩展事件流 | ：A1:信息格式不符  重新输入信息 |

|  |  |
| --- | --- |
| 用例编号 | UC03 |
| 用例名称 | 删除程序员 |
| 简要说明 | 管理员删除程序员 |
| 执行者 | 管理员 |
| 前置条件 | 管理员已成功登录 |
| 后置条件 | 管理员成功修改程序员信息 |
| 基本事件流 | 1.管理员进入管理员功能页面  2.管理员选择删除程序员功能  3.系统提示选择程序员  4.管理员点击删除  5.系统提示删除成功  6.用例结束 |
| 扩展事件流 | 无 |

|  |  |
| --- | --- |
| 用例编号 | UC04 |
| 用例名称 | 查询程序员信息 |
| 简要说明 | 管理员查询程序员信息 |
| 执行者 | 管理员 |
| 前置条件 | 管理员已成功登录 |
| 后置条件 | 管理员成功查询程序员信息 |
| 基本事件流 | 1.管理员进入管理员功能页面  2.管理员选择查询程序员信息功能  3.系统提示输入查询信息  4.管理员输入查询信息  A1:查询信息有误  查询失败  5.查询成功  6.用例结束 |
| 扩展事件流 | 无 |

（每一章都必须另起一页）

# 2 系统设计

## 2.1 多层结构设计

说明如何进行分层，各层运用哪些技术，各层分别由哪些package、页面以及其他资源构成，画出系统的组件图

The Presentation Layer is implemented as a Java EE Web Application and provides a very thin and concise Model-View-Controller type user interface to the Business and Persistence Layers. …

Since the PetClinic application is all about database access and there is very little business logic in the application outside of that, there is no separation of the primary Business and Persistence Layer API's. While this design technique should not be used for an application with more complex business logic, it is acceptable here because all of the non-persistence related business rules have been implemented in business objects and have not leaked into the Persistence Layer. The most important facet of the design is that the Business and Persistence Layers are COMPLETELY independent of the Presentation Layer. …

The Persistence Layer can be configured to use either HSQL or MySQL with any one of the following data access technologies aided by infrastructure provided by Spring:

* JDBC
* ……

本系统利用Spring MVC结构，用于替代servlet来响应请求，获取表单参数。分为三层M（model）层、V（view）层和C（controller）层，M层指repository和service；V层指的是页面，例如jsp/html等；C层指的是控制器， 每次客户端与服务器交互都要经过C层。首先请求发送request请求到C层，然后控制器到M层处理，处理结果再返回C层，而C层要经过V层渲染，最后返回终端（response）

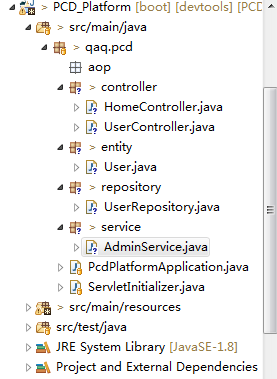


图2.1-1 项目工程目录

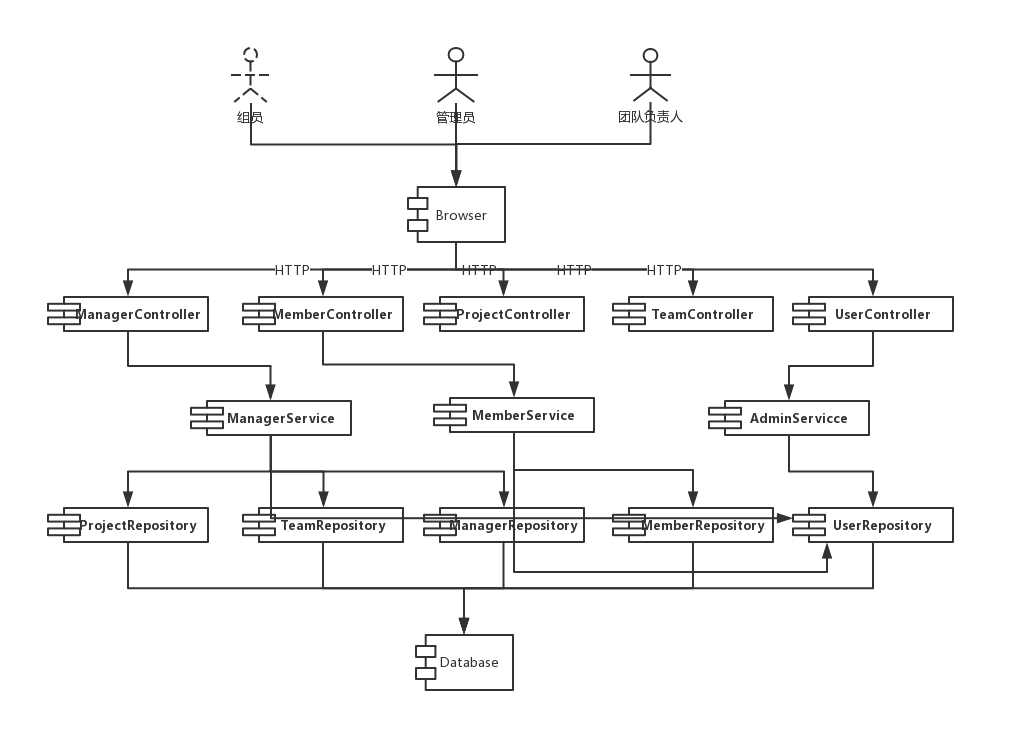


图2.1-2 项目协作开发平台

## 2.2 问题域设计

### 2.2.1 域模型设计

分析系统中的实体并给出系统的领域模型

以项目协作开发平台来说，项程序员、团队、项目、组员、团队负责人都是经常出现的名词，它们也都是从需求描述中识别出来的实体。团队和程序员是一对多的隶属关系，程序员和项目是一对多的拥有关系，团队与项目也是一对多的拥有关系，而程序员则泛化为组员和团队负责人



图2.2-1 项目协作开发平台域模型

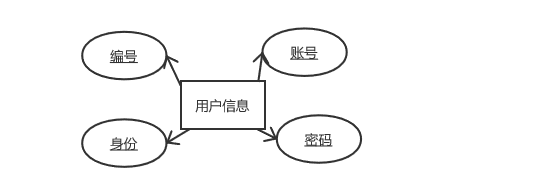


图2.2-2 用户PDM图

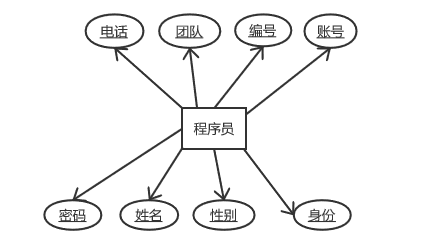


图2.2-3 程序员PDM图

### 2.2.2 Service接口设计

通过活动图、状态图、通信图或顺序图分析得到Service接口的方法，并画出相关类图。

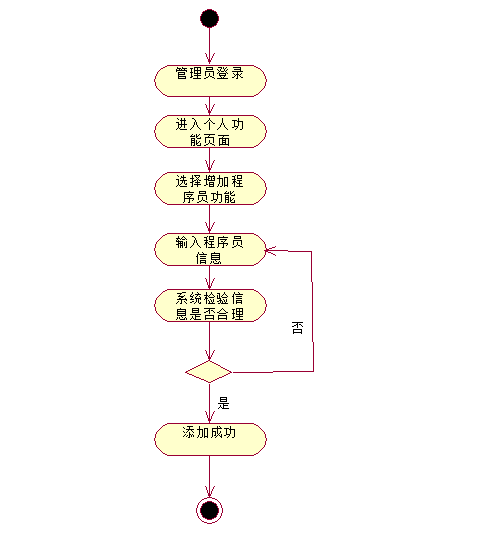


图2.2-4添加程序员活动图

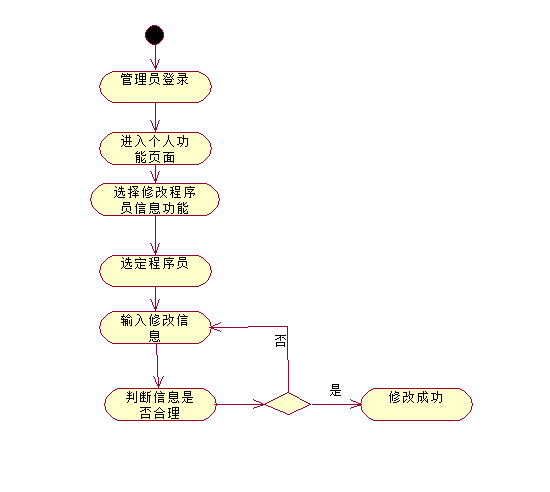


图2.2-5 修改程序员信息活动图

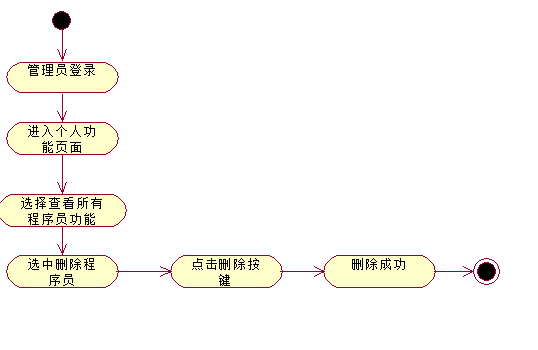


图2.2-6 删除程序员活动图

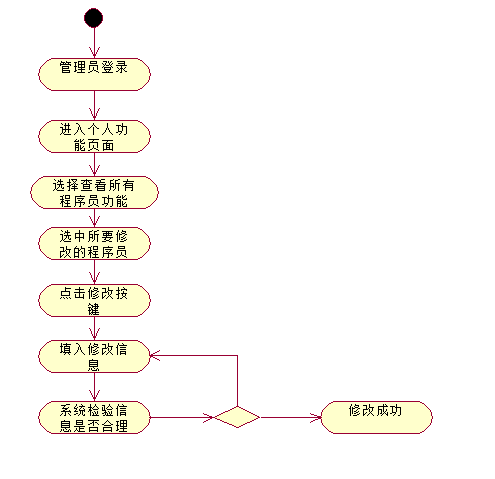


图2.2-7 修改程序员活动图

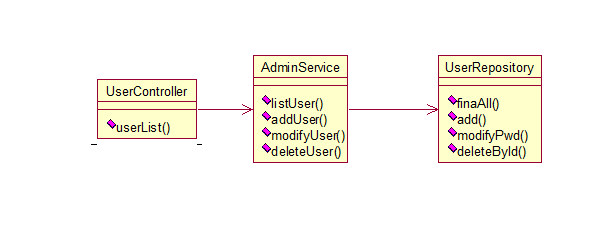


图2.2-8 管理员service类图

## 2.3 持久化设计

### 2.3.1 数据库设计

说明给出的数据库类型，并以表格形式给出数据库表结构

本系统运用Mysql数据库

表2.3-1 用户信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型（精度） | 字段描述 | 其他 |
| U\_id | NUMBER(9) | 用户编号 | PK（主键） |
| U\_account | VARCHAR(50) | 账号 | Null |
| U\_password | VARCHAR(50) | 密码 | Null |
| U\_status | CHAR(50) | 身份 | Null |

表2.3-2 程序员个人信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型（精度） | 字段描述 | 其他 |
| C\_id | NUMBER(9) | 程序员编号 | PK（主键） |
| C\_name | VARCHAR(20) | 名字 | Null |
| C\_account | VARCHAR(50) | 账号 | Null |
| C\_password | VARCHAR(50) | 密码 | Null |
| C\_sex | CHAR(10) | 性别 | “男”“女” |
| C\_tel | CHAR(50) | 联系方式 | Null |
| C\_team | CHAR(50) | 所在团队 | Null |
| C\_status | CHAR(50) | 程序员身份 | Null |

表2.3-3 团队信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型（精度） | 字段描述 | 其他 |
| T\_id | NUMBER(9) | 团队编号 | PK（主键） |
| T\_name | CHAR(50) | 团队名称 | Null |
| T\_mumber1 | CHAR(50) | 团队成员1 | Null |
| T\_mumber2 | CHAR(50) | 团队成员2 | Null |
| T\_mumber3 | CHAR(50) | 团队成员3 | Null |
| P\_name | CHAR(50) | 团队项目 | Null |

表2.3-4 项目信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型（精度） | 字段描述 | 其他 |
| P\_id | NUMBER(9) | 项目编号 | PK（主键） |
| P\_name | CHAR(50) | 项目名称 | Null |
| P\_beginTime | CHAR(50) | 开始时间 | Null |
| P\_overTime | CHAR(50) | 结束时间 | Null |
| P\_content | WORD | 项目介绍 | Null |

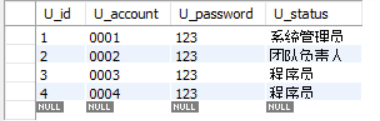


图2.3.1 用户表

### 2.3.2 数据访问设计

分析并给出数据库配置以及repository类的类图。

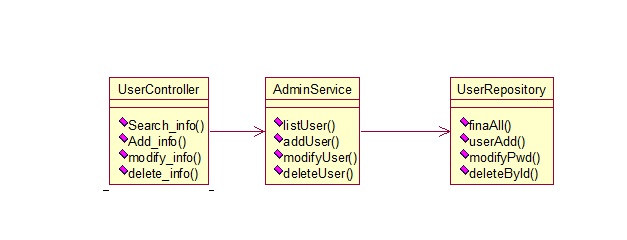


图2.3-2 用户repository类图

## 2.4 界面设计

### 2.4.1 页面链接关系

（分析页面之间的连接关系以及url所对应的Controller方法。）

表2.4-1 URL对应功能说明（管理员）

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP verb | URL | action（Controller类的方法） | used for |
| GET | templates\_2/add\_info | Add\_info | create a new programmer |
| POST | templates\_2/search\_info | Search\_info | query users |
| GET | templates\_2/modify\_info | Modify\_info | return an HTML form for editing a user’s password |
| POST | templates\_2/delete\_info | Delete\_info | delete a specific user |

The models will look like this:

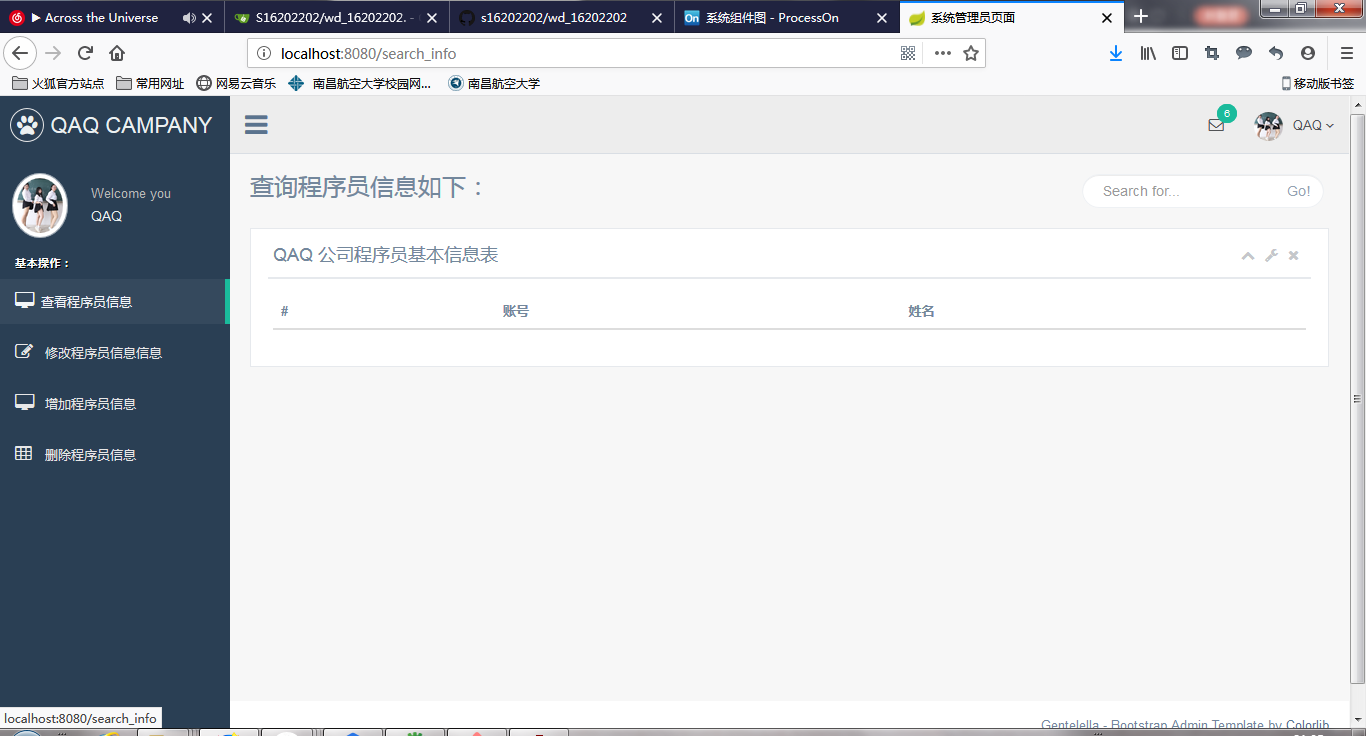
### 2.4.2 页面设计

（给出页面设计及其操作流程。）

You can then access petclinic here: http://localhost:8080/



图2.4-1 项目协作开发平台-管理员



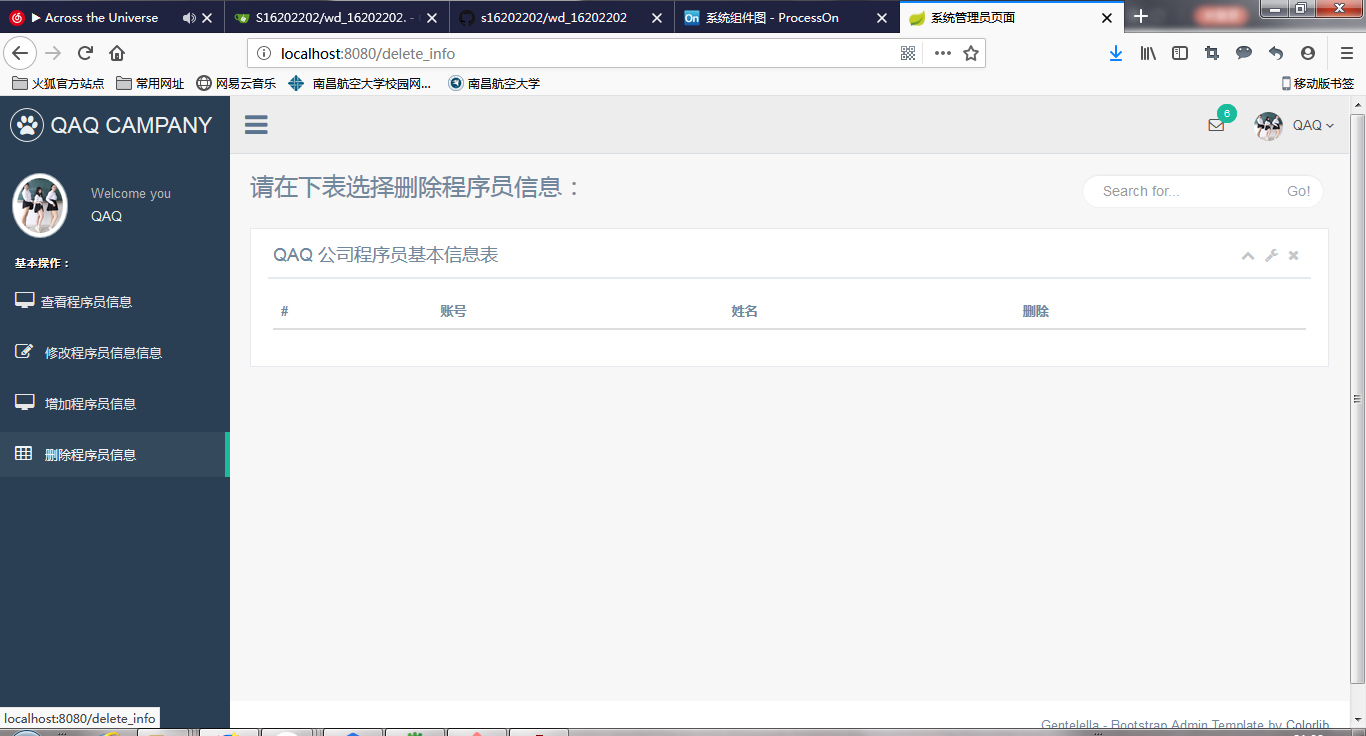
# 图2.4-2 项目协作开发平台-管理员查看

# 

# 图2.4-3 项目协作开发平台-管理员修改

# 

# 图2.4-3 项目协作开发平台-管理员添加



# 图2.4-3 项目协作开发平台-管理员删除

# 

# 3 系统实现

## 3.1 系统开发环境

给出页面设计及其操作流程。

A copy of the Spring runtime library jar file is provided with the sample application along with some of the other required jar files. The developer will need to obtain the following tools externally, all of which are freely available:

* Java SDK 1.5.x
* Ant 1.7.x
* Tomcat 5.x.x, or some other web application container
* JUnit 4.4 - needed to run the tests
* (Optional) MySQL 5.x with MySQL Connector/J 5.x

## 3.2 表达层

给出表达层文件结构，配置说明，并阐述个人模块内表达层的实现过程（含html、css、js、controller等，如有拦截器、配置类、校验类也一并说明）。

The PetClinic web application is configured via the following files:

* war/WEB-INF/web.xml: the web application configuration file.
* war/WEB-INF/petclinic-servlet.xml: configures the petclinic dispatcher servlet and the other controllers and forms that it uses. The beans defined in this file reference the Business/Persistence Layer beans defined in applicationContext-\*.xml.
* war/WEB-INF/classes/messages\*.properties: configures the definition of internationalizable message resources.

Presentation Layer classes

* org.springframework.samples.petclinic.web.ClinicController

is an annotation-driven, POJO MultiActionController that is used to handle simple display-oriented URLs.

* org.springframework.samples.petclinic.web.FindOwnersForm

is an annotation-driven, POJO Form controller that is used to search for Owners by last name.

* org.springframework.samples.petclinic.web.AddOwnerForm

is an annotation-driven, POJO Form controller that is used to add a new Owner to the system.

* org.springframework.samples.petclinic.web.EditOwnerForm

 is an annotation-driven, POJO Form controller that is used to edit an existing Owner. A copy of the existing Owner is used for editing.

* org.springframework.samples.petclinic.web.AddPetForm

is an annotation-driven, POJO Form controller that is used to add a new Pet to an existing Owner.

* org.springframework.samples.petclinic.web.EditPetForm

is an annotation-driven, POJO Form controller that is used to edit an existing Pet. A copy of the existing Pet is used for editing.

* org.springframework.samples.petclinic.web.AddVisitForm

is an annotation-driven, POJO Form controller that is used to add a new Visit to an existing Pet.

## 3.3 业务层

给出业务层文件结构，配置说明，并阐述个人模块内业务层的实现过程（含Service接口及其实现类等，如有Aspect类、日志类、自添加功能类也一并说明）。

The Business Layer consists of a number of basic JavaBean classes representing the application domain objects and associated validation objects that are used by the Presentation Layer. The validation objects used in PetClinic are all implementations of the org.springframework.validation.Validator interface.

* org.springframework.samples.petclinic.Entity is a simple JavaBean superclass used for all persistable objects.
* org.springframework.samples.petclinic.NamedEntity is an extension of Entity that adds a name property.
* org.springframework.samples.petclinic.Specialty is an extension of NamedEntity.
* org.springframework.samples.petclinic.PetType is an extension of NamedEntity.
* org.springframework.samples.petclinic.Person is an extension of Entity that provides a superclass for all objects that implement the notion of a person.
* org.springframework.samples.petclinic.Vet is an extension of Person that implements a veterinarian. It holds a List of specialties that the Vet is capable of.
* org.springframework.samples.petclinic.Owner is an extension of Person that implements a pet owner. It holds a List of pets owned.
* org.springframework.samples.petclinic.Pet is an extension of NamedEntity that implements a pet. It holds a List of visits made concerning the pet.
* org.springframework.samples.petclinic.Visit is a simple JavaBean that implements the notion of a clinic visit for a pet.
* org.springframework.samples.petclinic.util.EntityUtils provides utility methods for handling entities.
* org.springframework.samples.petclinic.validation.OwnerValidator is a Spring Validator that verifies correct data entry for the Add and Edit Owner forms.
* org.springframework.samples.petclinic.validation.PetValidator is a Spring Validator that verifies correct data entry for the Add and Edit Pet forms.
* org.springframework.samples.petclinic.validation.VisitValidator is a Spring Validator that verifies correct data entry for the AddVisit form.

## 3.4 持久化层

给出业务层文件结构，数据库配置说明，并阐述个人模块内业务层的实现过程（含repository、类等，如有网络、文件访问类也一并说明）。

A Spring org.springframework.context.ApplicationContext object provides a map of user-defined JavaBeans that specify either a singleton object or the initial construction of prototype instances. These beans constitute the Business/Persistence Layer of PetClinic. The following beans are defined in all 3 versions (1 per access strategy) of the PetClinic war/WEB-INF/applicationContext-\*.xml file:

* A PropertyPlaceholderConfigurer, which is configured via <context:property-placeholder … /> and is a singleton bean that replaces ${…} placeholders with values from a properties file, in this case, JDBC-related settings for the dataSource bean described below (see src/jdbc.properties).
* dataSource, which is a singleton bean that defines the implementation of the source of database connections used by the application.
* transactionManager, which is a singleton bean that defines the implementation of the transaction management strategy for the application.
* clinic, which is a singleton bean that defines the implementation of the Clinic interface that provides the primary Business Layer API of the application.

# 4 系统测试

给出测试用例，说明测试方法与测试过程，并阐述个人模块内单元测试与功能测试的完成过程（测试过程需截图，同时并给出测试结果）。

AbstractClinicTests and its subclasses benefit from the following services provided by the Spring TestContext Framework

* org.springframework.samples.petclinic.OwnerTests

is a simple JUnit 4 based TestCase that supports Business Rule #1.

* org.springframework.samples.petclinic.AbstractClinicTests

is a JUnit 4 based TestCase requiring a live database connection that is used to confirm correct operation of the database access objects in the various implementations of the Clinic interface. "AbstractClinicTests-context.xml" declares a common javax.sql.DataSource. Subclasses specify additional context locations which declare a org.springframework.transaction.PlatformTransactionManager and a concrete implementation of Clinic.

* AbstractClinicTests extends AbstractTransactionalJUnit4SpringContextTests, one of the valuable testing support classes provided by the Spring TestContext Framework found in the org.springframework.test.context package. The annotation-driven configuration used here represents best practice for integration tests with Spring. Note, however, that AbstractTransactionalJUnit4SpringContextTests serves only as a convenience for extension. For example, if you do not wish for your test classes to be tied to a Spring-specific class hierarchy, you may configure your tests with annotations such as @ContextConfiguration, @TestExecutionListeners, @Transactional, etc.

# 5 总结与体会

不少于1500字，说明系统的特色，分析系统的不足，给出今后的改进方向，最后谈谈自己的体会。