大学生问答社区

软件架构文档

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 08/11/20 | 1.0 | 完善各节内容 | 胡昊源、施超、王兴宇、熊天磊 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 参考资料 4

2. 用例视图 4

3. 逻辑视图 4

3.1 概述 4

3.2 在构架方面具有重要意义的设计包 4

4. 进程视图 4

5. 部署视图 5

6. 实现视图 5

7. 数据视图 5

8. 技术视图 6

9. 核心算法设计 7

10. 性能与质量属性 7

软件架构文档 （简化版）

# 简介

## 目的

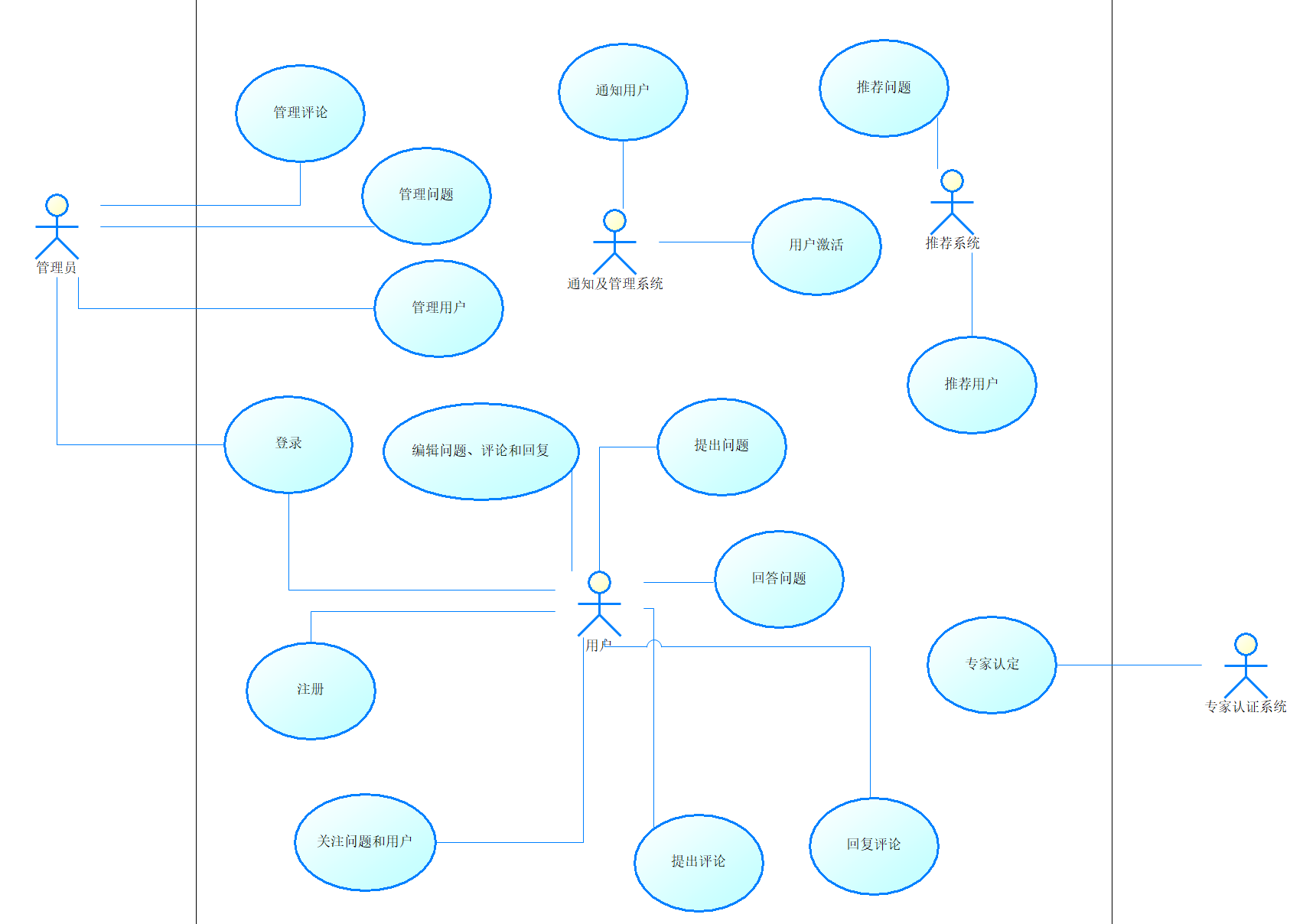
本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架视图来描述系统的各个方面。它用于记录并表述已对系统的构架方面做出的重要决策。

本文档记录了软件架构的各种视图，包括用例视图、逻辑视图、进程视图、部署视图、实现视图、数据视图和技术视图。本文档同时还简要介绍了一些核心算法设计。

## 参考资料

《软件工程原理》，ISBN 978 -7 -04 -036906 -9，沈备军、陈昊鹏、陈雨亭，2013.2，北京高等教育出版社

# 用例视图



1用例图

本软件架构中主要包含以下用例：

用户注册：创建一个新用户账户。

用户登录：已注册账户登录入系统。

提出问题：用户在问答社区中创建一个新问题。

回答问题：用户在问答社区中对已有问题新建一个回答。

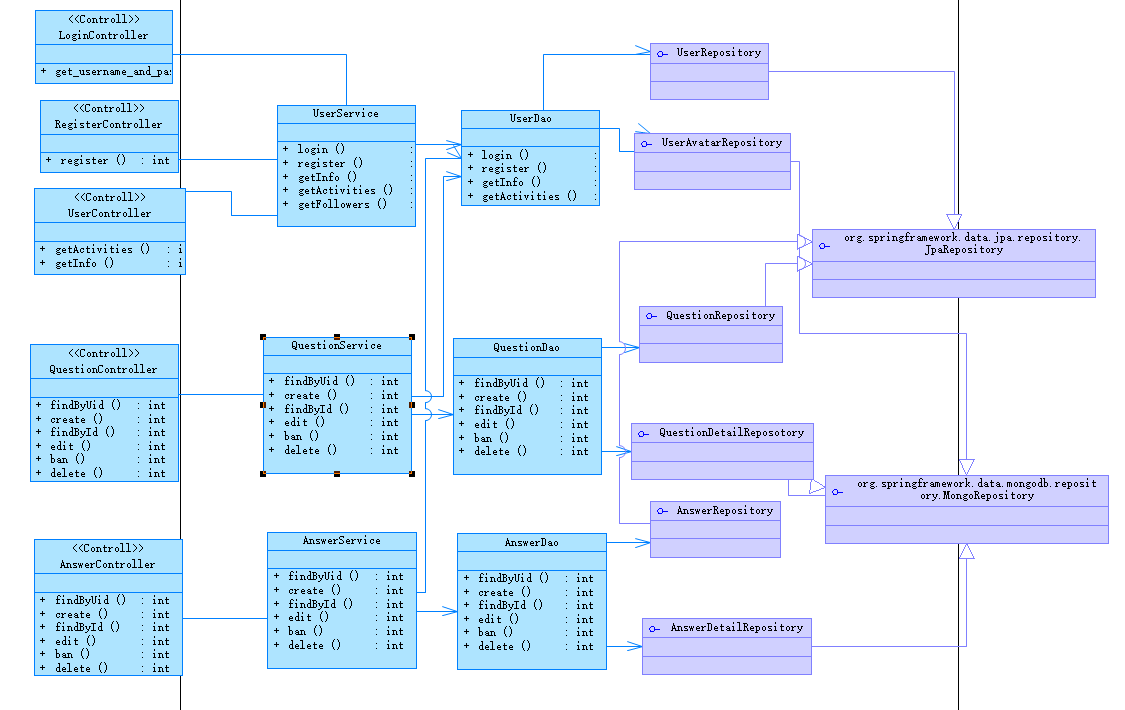
评论、回复评论：用户对回答发表评论，或对已有评论进行评论。

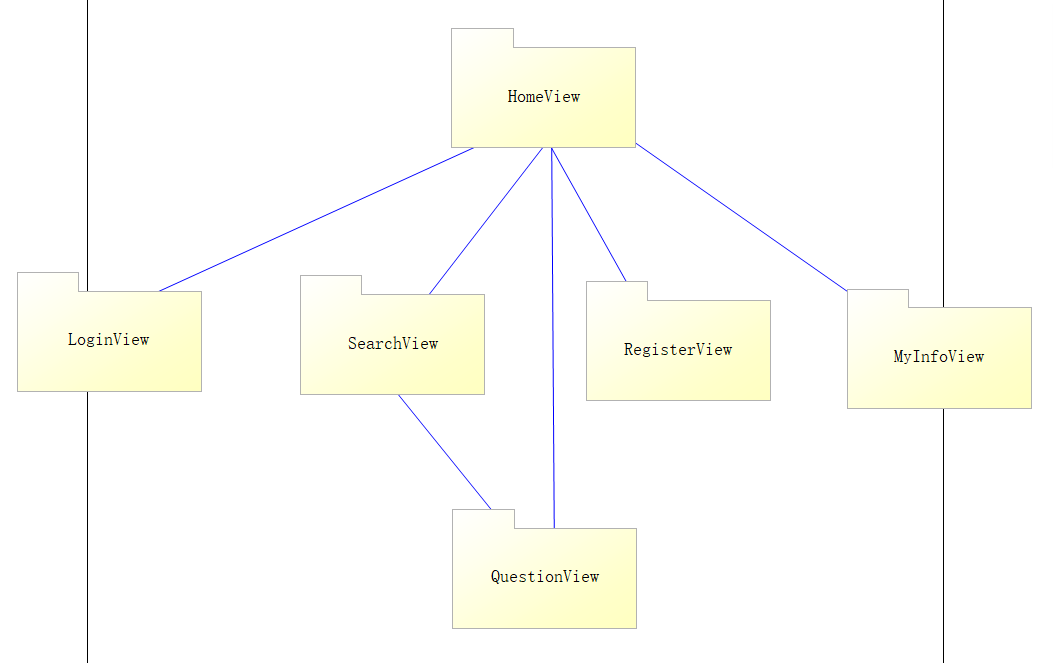
# 逻辑视图

QAQ大学生社区问答系统的逻辑视图由三个主要的包组成：用户界面、业务服务和业务对象。

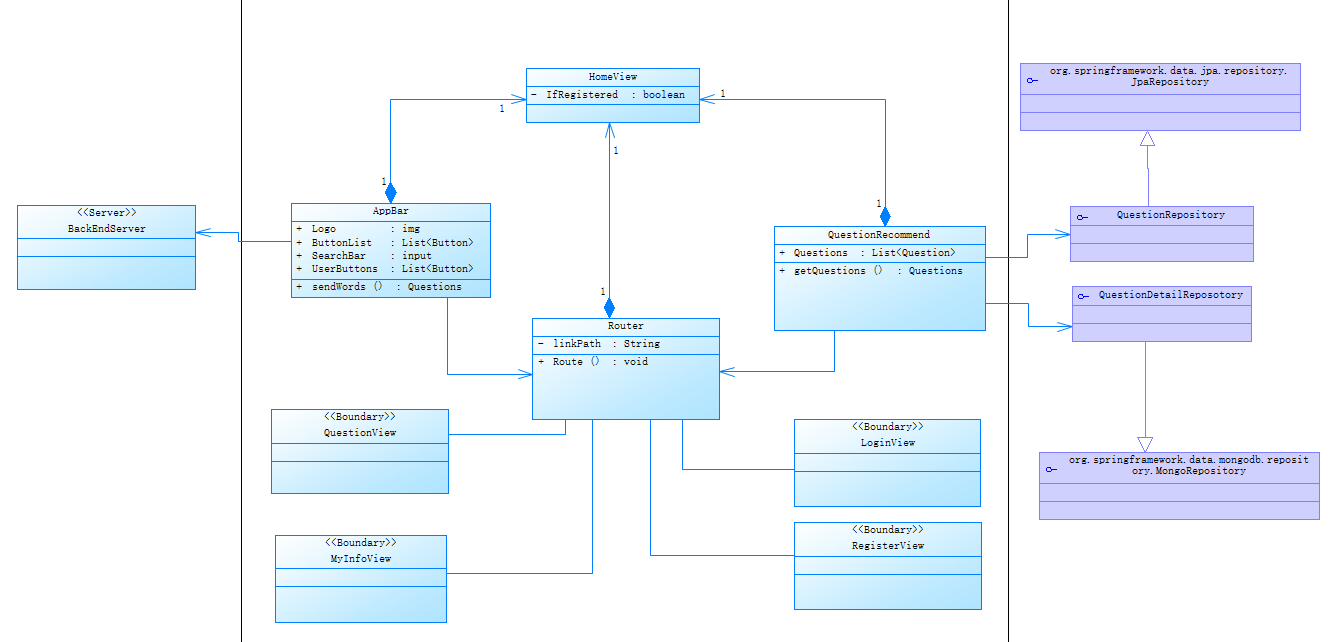
用户界面包含用户与系统通信的每个表单的类。逻辑视图中描述了用户界面的组成，并对前端主页的架构进行了详细的分析。

业务服务包含系统对用户对应的操作所提供的各种服务。逻辑视图中描述了业务服务的组成，并对核心业务：问题/答案/用户的搜索、修改、增加、删除进行了详细的分析。如下图所示。后端遵循MVC架构，并在此基础上进一步细化，将结构划分为了Controller, Service, Data Access Object, Repository各层。其中Controller层负责对处理用户请求，Service层完成各种请求。DAO层负责向上层提供数据。Repository层负责从数据库获取数据。

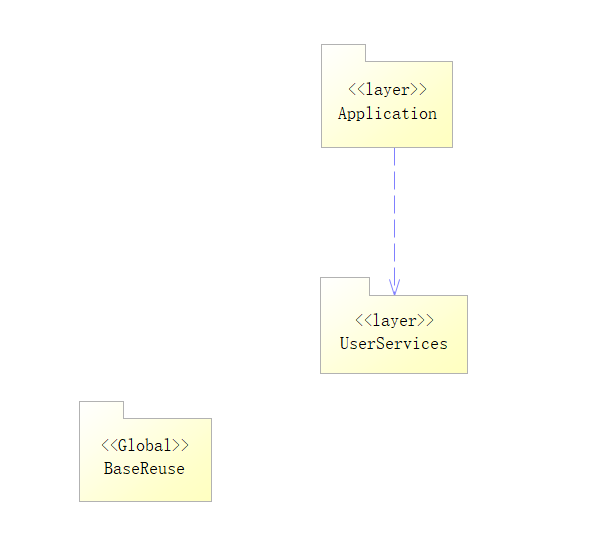




2前端页的架构视图



3前端主页的逻辑视图



4逻辑视图总架构（部分）

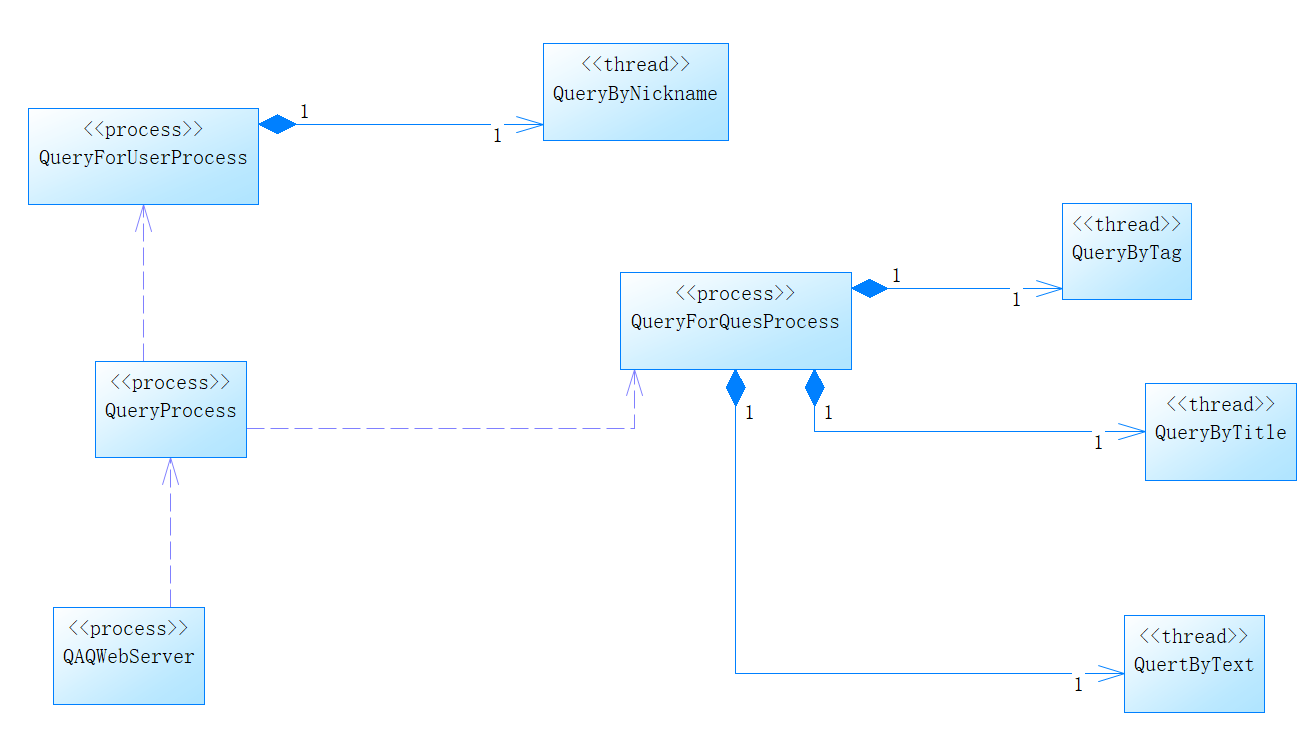
应用层：这个应用层有所有的边界类，它们代表用户看到的应用程序屏幕。

用户服务层：用户服务流程层有所有的控制器类，它们表示驱动应用程序行为的用例管理器。此层表示客户端到中间层的边界。

基本重用层：基本重用层包括支持列表函数和模式的类。

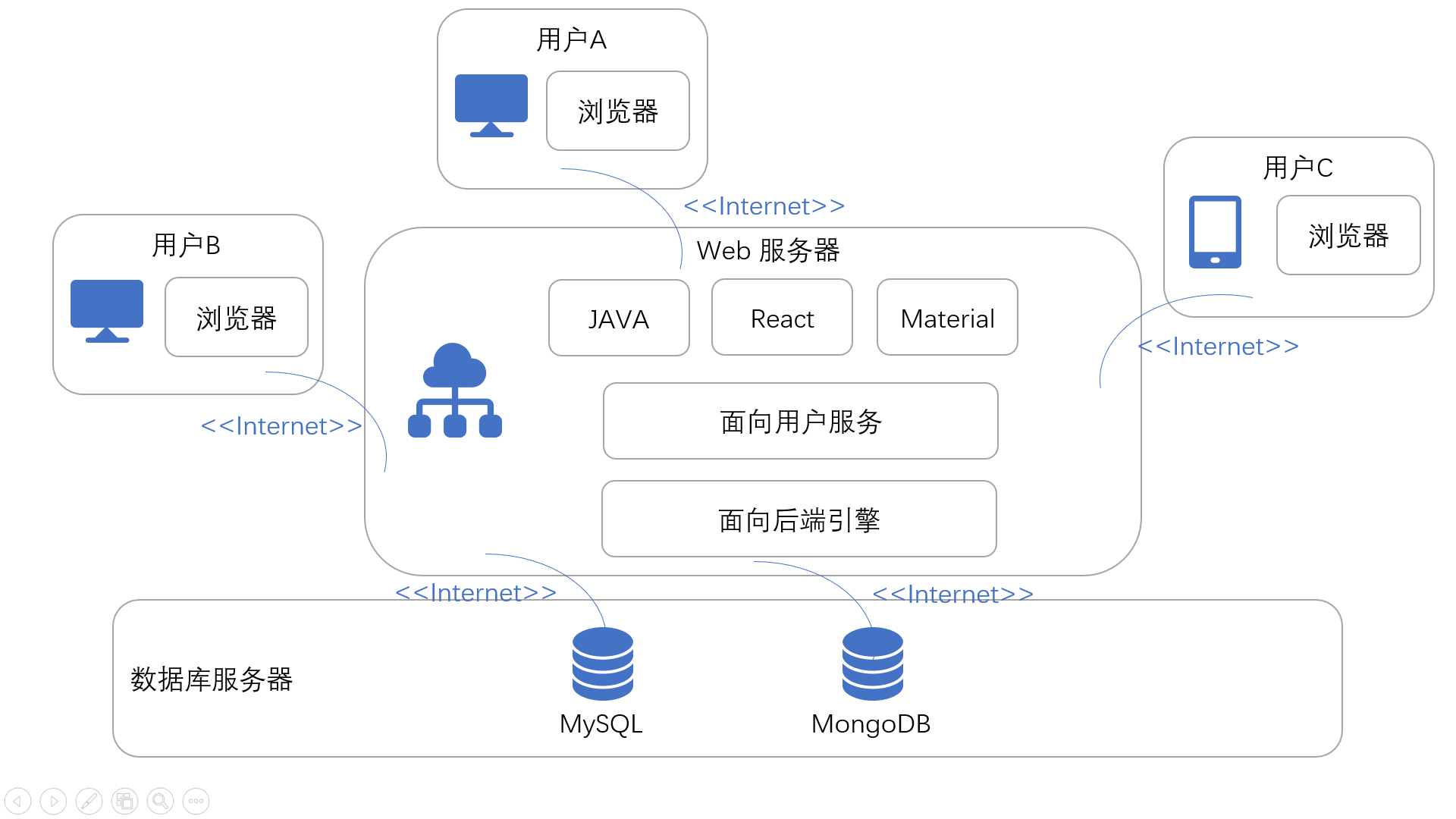
# 进程视图

在该软件架构中，较为核心的进程与搜索功能相关。针对搜索功能的进程进行绘图，用户运行服务器后，可以搜索用户，也可以搜索问题。搜索用户时会根据nickname参数对数据库进行访问，而搜索问题时，会根据标签（tag），标题（title）以及内容（text）进行搜索。



# 部署视图

QAQ大学生问答社区的部署视图描述了最典型的平台配置的各种物理节点，同时也描述了任务从流程图到物理节点的分配。部署视图按物理网络配置进行组织，每个这样的配置都由一小块部署图来表示。部署图同时也表示了进程到每个处理器的映射。



5QAQ大学生问答社区的部署视图

1. **用户端（桌面端/移动端）**

用户生使用外部浏览器访问网页，这些终端通过以太网连接到Web服务器。

1. **Web服务器**

Web服务器部署在谷歌云端，所有用户都可以通过以太网访问服务器。面向用户时，服务器提供所支持的服务，如搜索问题、回答问题等。面向后端数据库时，服务器负责进行对数据进行增、删、改、查等操作。

1. **数据库服务器**

数据库服务器中包括MySQL数据库和MongoDB数据库，分别负责存储关系型数据和非关系型数据，为Web服务器提供必要的数据信息。

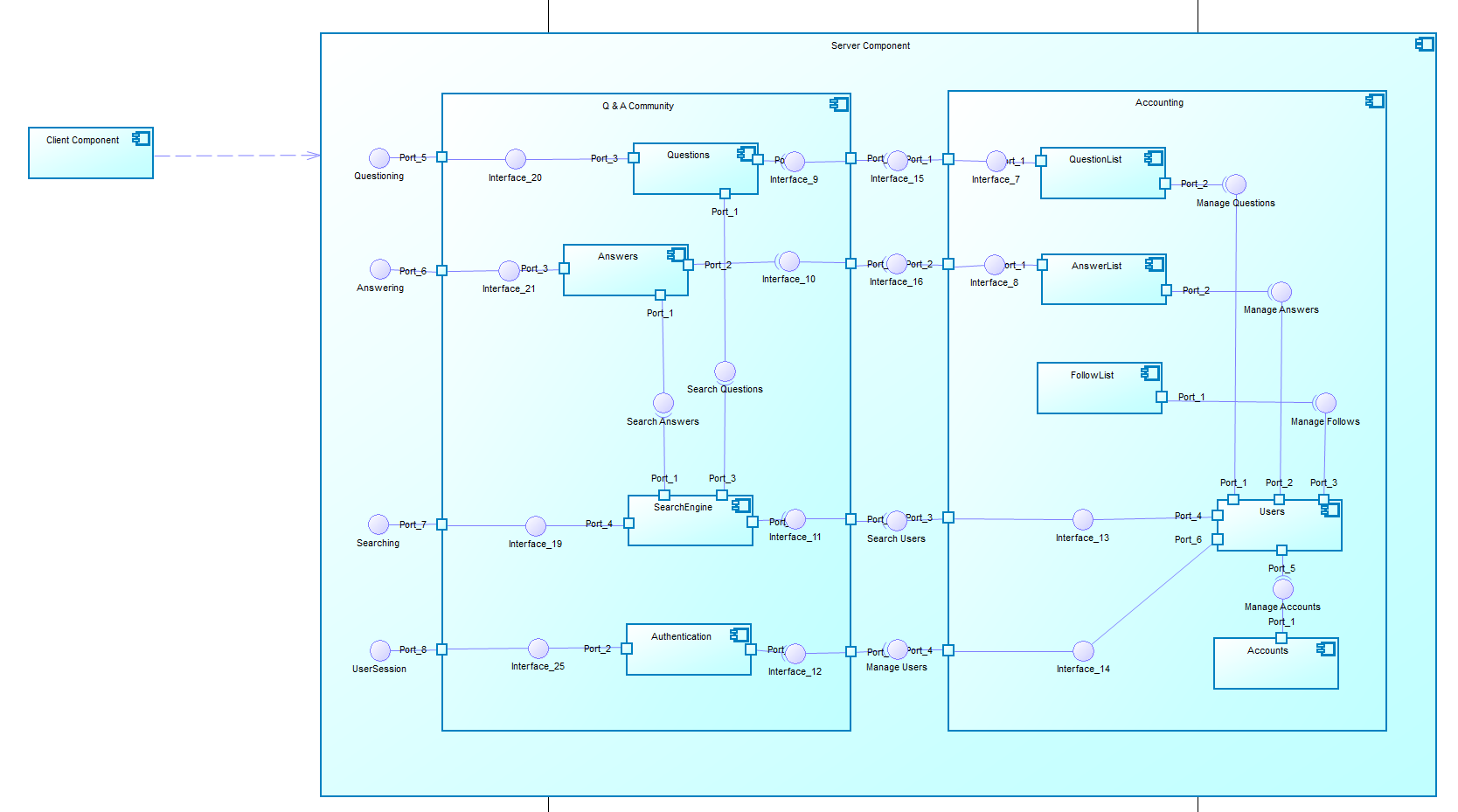
# 实现视图

本节说明本架构的整体结构、软件分解为实现模型中的层和子系统的情况，以及所有在构架方面具有重要意义的构件。

Server部分主要包括两个部分：

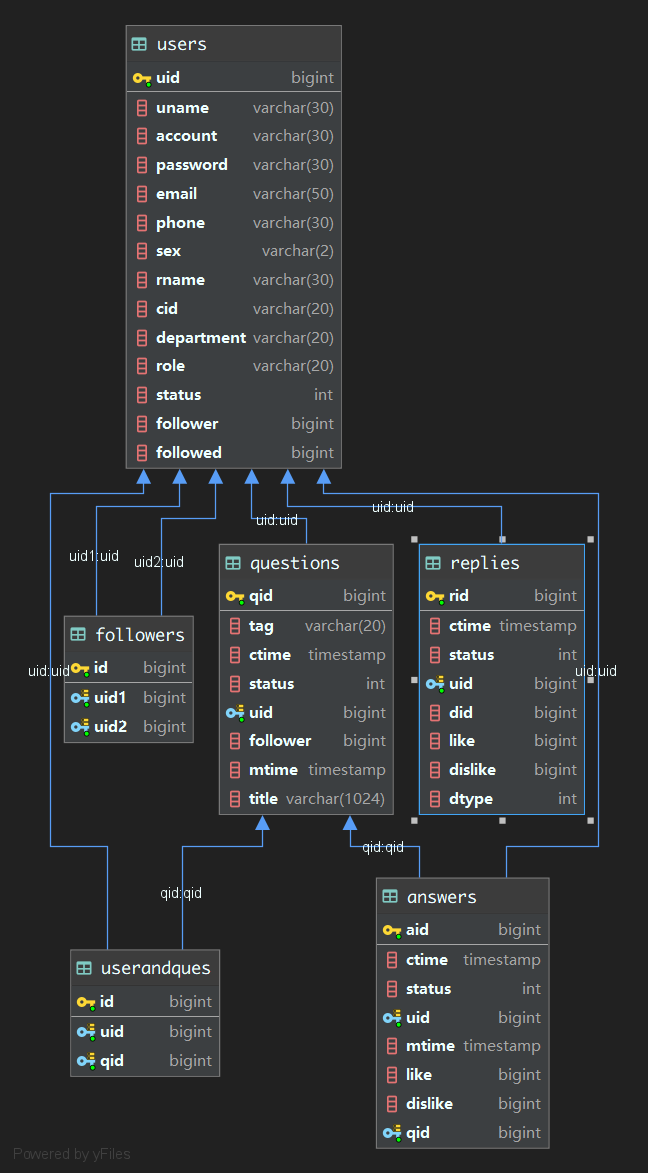
Q & A Community子系统：包含四个组件：Questions、Answers、SearchEngine、Authentication。Questions和Answers组件允许用户创建提问和编写回答，分别使用（Accounting子系统中）QuestionList和AnswerList组件提供的ManageQuestions和ManageAnswers界面；SearchEngine组件允许用户通过Searching界面搜索浏览问题、回答及用户；Authentication组件允许用户创建、登录或注销账户。

Accounting子系统：提供四个接口：ManageQuestions、ManageAnswers、SearchUsers、ManageUsers，连接到Q & A Community以实现相关功能。



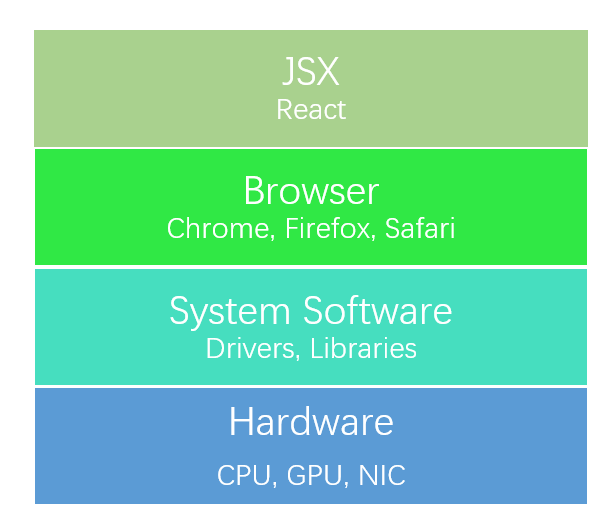
# 数据视图

该架构采用的数据库设计如下图所示。每个实体类均有主键。不用表之间的关联使用外键约束。

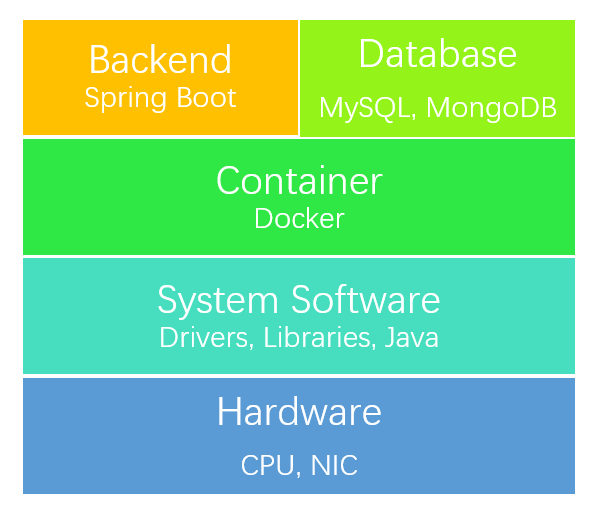


# 技术视图

以下的技术视图展示了本软件前端、后端的技术选型。

前端运行于浏览器中，用React作为框架。底层为相关硬件、系统软件。

后端技术视图底层为硬件。系统软件包括相关驱动、库和JRE。在系统软件层之上我们使用Docker来提供容器支持，从而运行后端和数据库。后端使用了Spring Boot 框架，遵循MVC框架进行开发。数据库使用了MySQL及MongoDB.



# 核心算法设计

系统中的核心算法包括搜索算法的推荐算法。

搜索算法基于用户输入的关键字，对全数据的内容进行搜索，包括标题、正文、标签等。如果用户输入的搜索内容较长，会进行分词并搜索。

推荐算法采用潜在因子算法。

# 性能与质量属性

该软件架构可以支持以下关键的性能需求：

1. 系统能够支持在有1000个并发用户的情况下，在3秒内完成响应。
2. 系统能够在2分钟内完成超过80%的事务。
3. 客户端不需要安装独立的客户端，只需要任何一款受兼容的浏览器即可运行。

该软件架构通过 client-server 架构完成上述的非功能性需求。客户端运行在浏览器中，通过HTTP协议与服务器端交互。

该架构同时支持以下的质量需求：

1. 客户端浏览器兼容超过全球统计数据中99.8%的浏览器版本，not\_dead 中所定义的浏览器版本。
2. 用户界面设计对用户友好，能够让会使用电脑的用户轻松使用。
3. 系统的可用性保证在99.9%以上。
4. 系统的Mean Time Between Failures (MTBF)应当超过300小时
5. 用户始终能通过互联网获取到最新的客户端，而无需关心版本更新的细节