教学辅助平台

软件架构文档

版本 <1.2>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 6/11/2022 | 1.0 | 编写简介和用例视图 | 李耀 |
| 8/11/2022 | 1.1 | 编写逻辑视图、部署视图、实现视图和技术视图 | 陈浩 |
| 9/11/2022 | 1.2 | 编写数据视图和质量属性设计 | 陈俊咏 |
|  |  |  |  |

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 5

4. 部署视图 6

5. 实现视图 7

6. 技术视图 7

7. 数据视图 8

8. 质量属性的设计 8

软件架构文档

# 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作出的重要决策。

本文档的主要由用例视图、逻辑视图、部署视图、实现视图、技术视图、数据视图以及质量属性设计等部分组成，主要面向本系统的开发人员，用于系统的开发和测试。

## 参考资料

[1]沈备军. 软件工程原理. 高等教育出版社, 2013. Print. 高等学校软件工程系列教材.

[2]汉森. Learning TypeScript中文版 = Learning TypeScript. 电子工业出版社, 2016. Print.

[3]班克斯. React学习手册. 中国电力出版社, 2021. Print.

[4]黄索远. Django项目开发实战 = Django Project Development Practice. 清华大学出版社, 2020. Print.

[5]刘宇宙. Python实战之数据库应用和数据获取. 电子工业出版社, 2020. Print. 新工科建设之路·数据科学与大数据系列.

# 用例视图

本系统用例图如图2-1所示，本系统主要有3个actor：管理员、教师、学生。其中最重要的用例就是publish problem or solution，该用例包含前端对数据的读取，后端中控制器向题目或者题解视图读取数据并将数据传递给题目或题解模型进行存储；该用例还包含本系统对OCR模块的接口调用。

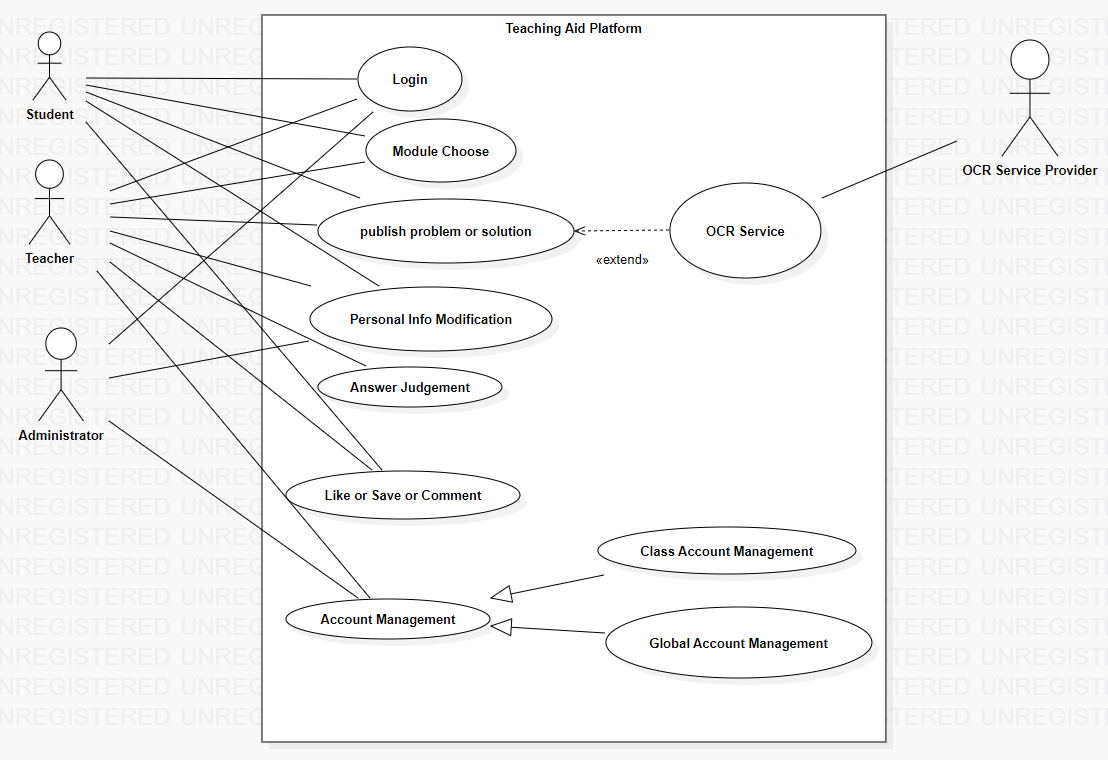


图2-1 系统用例图

# 逻辑视图

## 概述

本系统为B/S架构，Browser端和Server端都主要采用MVC架构，即把系统划分为Model层、Controller层和View层,其中Controller层是MVC框架的核心。在Brower端中，主要包括题目题解视图、用户信息视图、偏好视图和课程视图。也包括处理发布事件的控制器、消息通知控制器和修改用户信息控制器。在Browser端主要有UserInfo和ProblemList两个model.Sever端的Controller层经过抽象后主要有3个controller: problem controller、user controller和class controller,其中problem controller请求有关题目和题解的数据并将修改的数据传递给前端；user controller向前端发送和请求有关用户个人信息和对题解收藏点赞的数据并将修改的数据传递给user模型；class controller用于处理课程相关的业务逻辑。

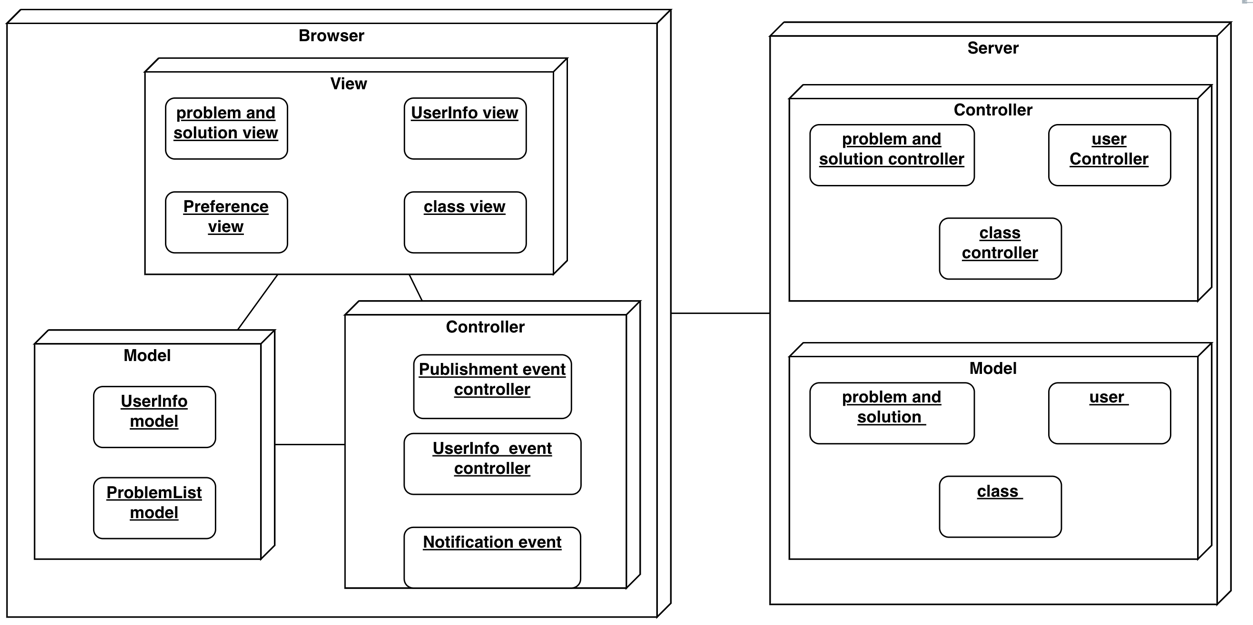


图3-1 逻辑视图

## 在构架方面具有重要意义的设计包

本系统经过抽象，可得5个包：课程管理包、题目与题解管理包、用户管理包、前端应用包和OCR服务包。本系统包图结构如图3-2所示：

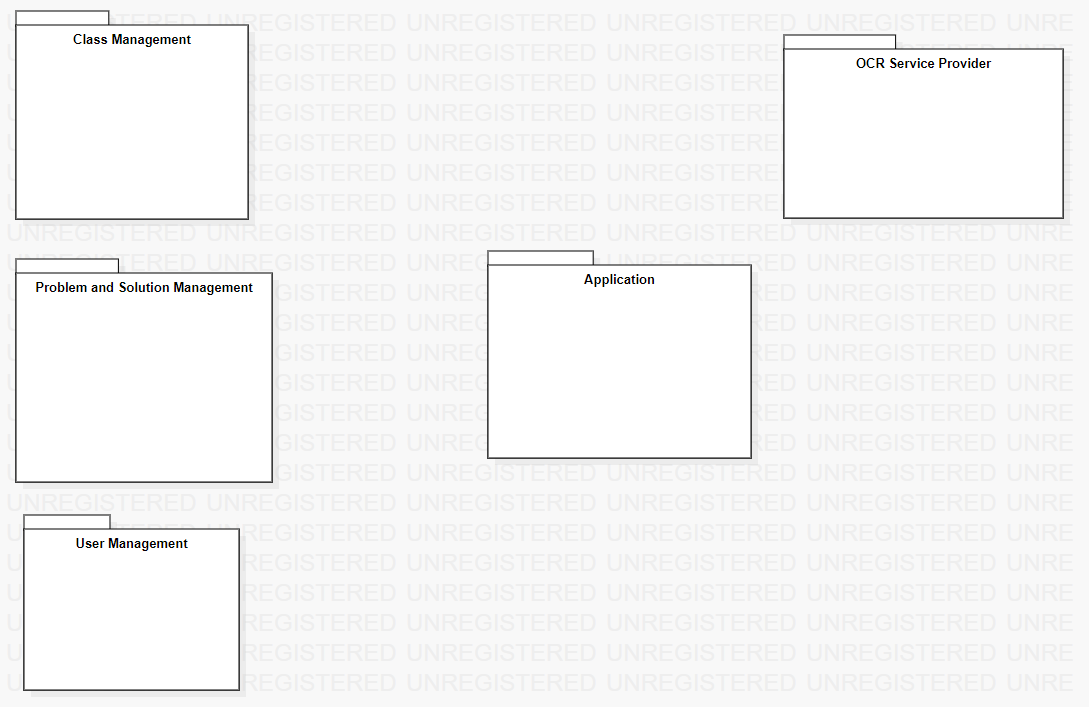


图3-2系统包图

此处主要介绍用户管理包中的重要类。用户管理包有3个实体类：学生、教师和管理员。其中学生和教师依赖于普通用户类，学生具有发布题目和题解以及发表评论的能力，教师在学生能力的基础上额外具有设置标准题解和管理课程用户的能力；管理员继承用户类并具有管理全局用户以及对课程模板管理的能力。

图示

描述已自动生成

图3-3 用户类图

# 部署视图

本系统是B\S架构，主要部署在一台服务器上面进行运行，通过浏览器进行访问和操作。系统同时通过调用第三方OCR服务实现相应的功能。

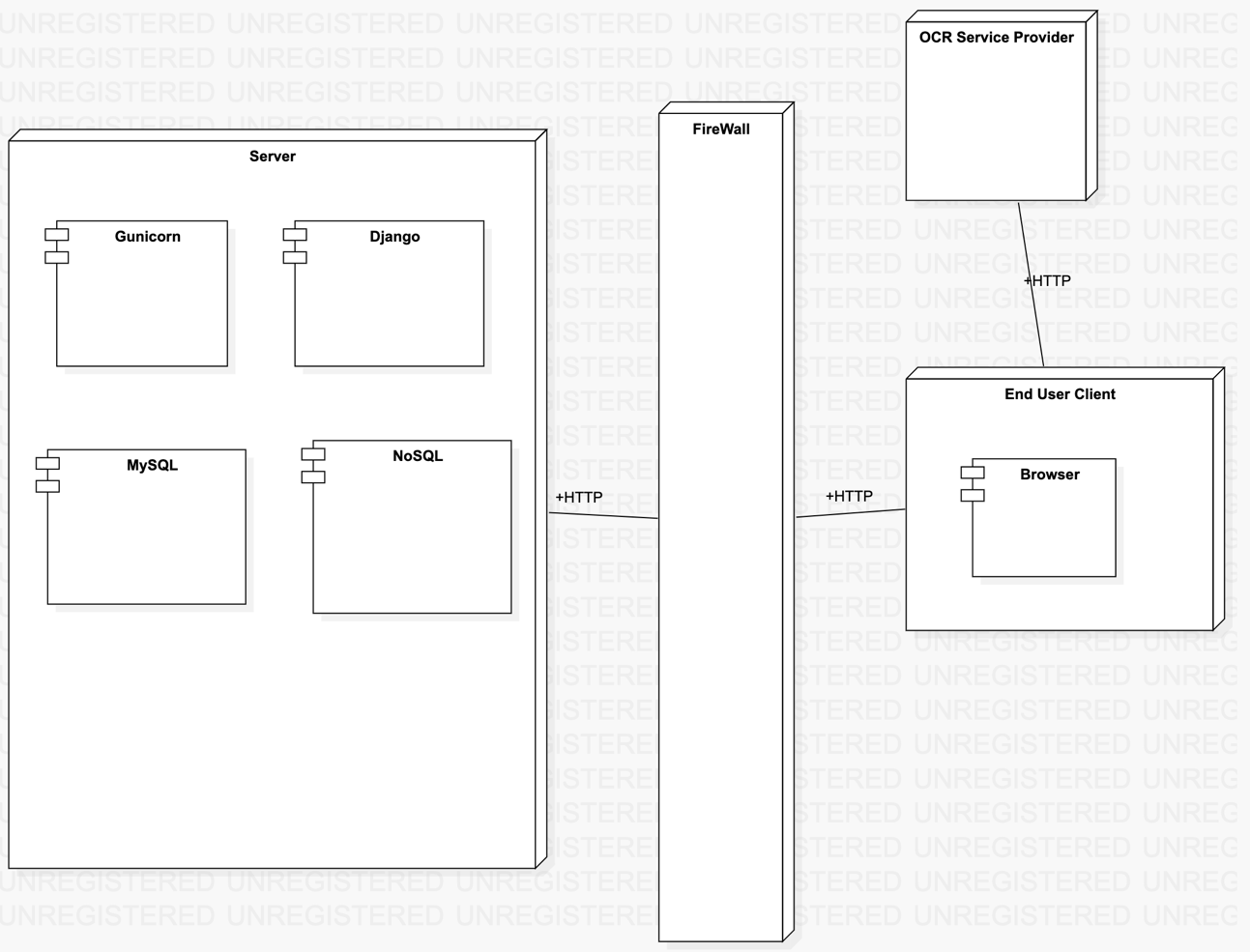


图4-1部署视图

# 实现视图

系统主要分为三大类功能：用户相关（User）、课程相关(Class)和题目题解相关(Publishment)。用户相关指用户的登录、修改信息等功能的实现。课程相关指课程的增删查改和模版的修改等功能的实现。题目题解相关指题目题解的发布和查看等相关功能的实现。

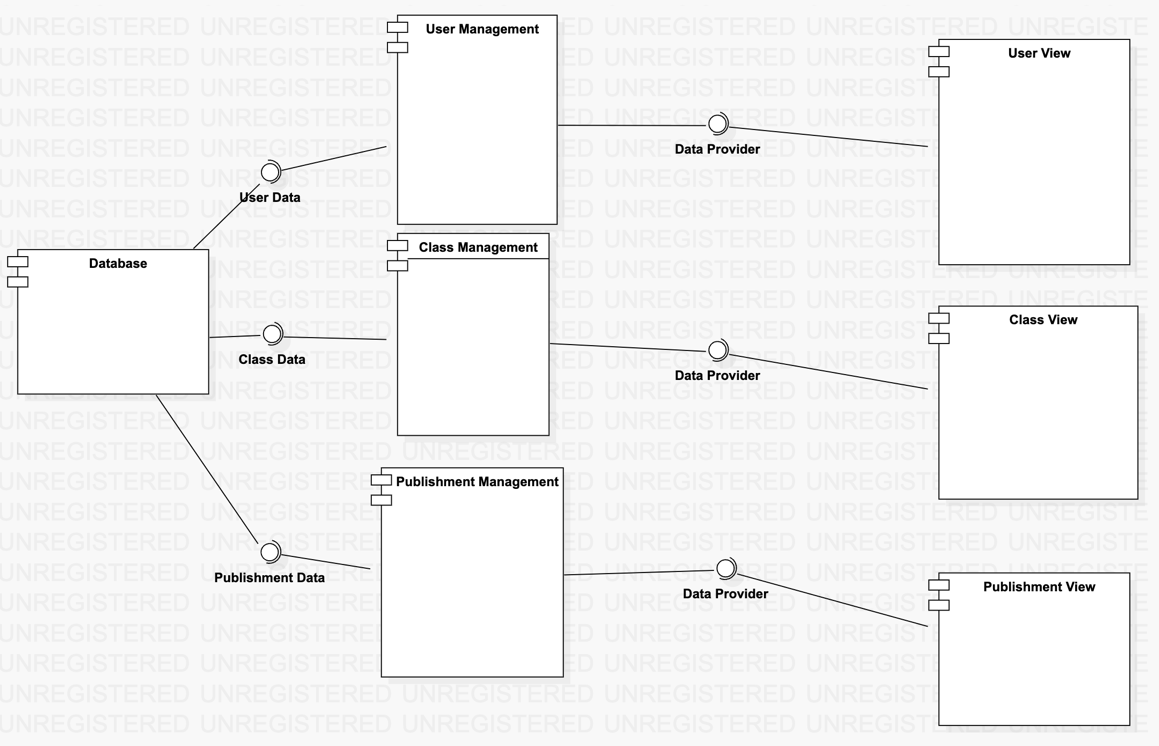


图5-1系统实现视图

# 技术视图

系统前端使用React框架和Ant Design pro进行开发，同时使用UMI和DVA等技术进行开发。后端主要使用Django框架和python语言进行开发，使用Gunicorn server；数据库使用Mysql和Nosql进行相关数据的存储。同时调用外部OCR服务。

图形用户界面, 图示, 应用程序

描述已自动生成

图6-1 技术视图

# 数据视图

本系统数据库e-r图主要有4张实体表，2张关系表。其中，用户表存储系统用户的个人信息及对应权限，以“用户id”作为主键；题目表存储课程题目的基本信息和发布题目的用户信息，以“题目id”为主键、“用户id”为外键；题解表存储题目答案的信息和题解对应的题目和发布者的信息。以“题解id”为主键、“题目id”“用户id”为外键；评论表存储用户对题解的评论信息和发布者的信息，以“评论id”为主键、“题解id”“用户id”为外键；点赞表是关系表，该表存储用户对评论的点赞情况，以“用户id”和“评论id”作为外键和主键；收藏表是关系表，该表存储用户对题解的收藏情况，以“用户id”和“题解id”作为外键和主键。

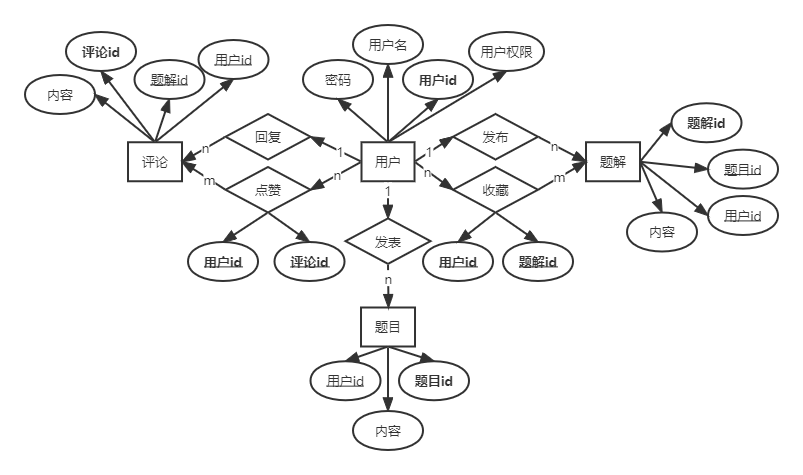


图7-1 系统数据库E-R图

# 质量属性的设计

（1）本系统采用的MVC架构可提高本系统的重用性、适用性、可扩展性和可维护性。MVC分层架构，是一种三层的架构，主要采用封装的思想来降低耦合度，从而使系统更灵活，扩展性更好。MVC模式允许用户使用各种不同样式的视图来访问同一个服务器端的代码，包括任何WEB（HTTP）浏览器或者无线浏览器（WAP），例如，很多数据可能用HTML来表示，但是也有可能用WAP来表示，而这些表示仅需要改变view层的实现方式，而controller层和model层无需做任何改变。分离视图层和业务层也使得WEB应用更便于维护和修改。

（2）本系统采用docker容器，将系统所需的环境配置在容器中可有效提高系统的可移植性与安全性。

（3）本系统前端采用react，用户可通过简单的操作调用后台封装的功能，可有效提高系统的易用性。

（4）系统采用日志管理技术，记录系统操作，能够在系统出现事故的时候进行恢复，提高了系统的可靠性。

（5）系统通过使用缓存技术在前后端缓存常用数据，减少与后端的交互；同时使用ajax异步通信，提高交互的效率，从而提高系统的性能。