

JIANGXI AGRICULTURAL UNIVERSITY

**本 科 毕 业 论 文（设 计）**



题目**： 国际半导体企业合作伙伴管理**

**系统的设计与开发**

**学 院： 软件学院**

**姓 名： 喻文鑫**

**学 号： 20171766**

**专 业： 软件工程**

**班 级： 软件1703班**

**指导教师： 吴沧海副教授**

**答辩日期： 2021 年4月17日**

目录

[摘要 I](#_Toc68308914)

[Abstract II](#_Toc68308915)

[1 引言 1](#_Toc68308916)

[1.1 课题研究的背景与意义 1](#_Toc68308917)

[1.2 国内外现状 1](#_Toc68308918)

[1.2.1 国内主要现状 1](#_Toc68308919)

[1.2.2 国外主要现状 2](#_Toc68308920)

[1.3 课题研究目的及主要内容 2](#_Toc68308921)

[1.3.1 课题研究目的 2](#_Toc68308922)

[1.3.2 课题研究的主要内容 2](#_Toc68308923)

[1.4 可行性分析 2](#_Toc68308924)

[1.4.1 技术可行性研究 3](#_Toc68308925)

[1.4.2 经济可行性研究 3](#_Toc68308926)

[1.4.3 操作可行性研究 3](#_Toc68308927)

[1.5 论文内容安排 3](#_Toc68308928)

[2 系统开发技术及环境 5](#_Toc68308929)

[2.1 编程语言Java 5](#_Toc68308930)

[2.2 集成开发环境idea 5](#_Toc68308931)

[2.3 数据库MySql 6](#_Toc68308932)

[2.4 RESTful接口 6](#_Toc68308933)

[2.5 应用服务器Tomcat 7](#_Toc68308934)

[2.6 系统开发环境Windows10 7](#_Toc68308935)

[3需求分析 9](#_Toc68308936)

[3.1 系统功能需求 9](#_Toc68308937)

[3.2 系统性能需求 10](#_Toc68308938)

[3.3 系统运行需求 11](#_Toc68308939)

[3.4 系统的用例分析 11](#_Toc68308940)

[4 国际企业合作伙伴管理系统的设计与开发 14](#_Toc68308941)

[4.1 概要设计 14](#_Toc68308942)

[4.1.1 系统架构设计 14](#_Toc68308943)

[4.1.2 数据库设计 14](#_Toc68308944)

[4.2 详细设计 18](#_Toc68308945)

[4.2.1 用户注册功能 18](#_Toc68308946)

[4.2.2 用户登录功能 19](#_Toc68308947)

[4.2.3 用户密码修改功能 20](#_Toc68308948)

[4.2.4 权限控制功能 21](#_Toc68308949)

[4.2.5 合作方操作功能 22](#_Toc68308950)

[4.2.6 管理员审核功能 24](#_Toc68308951)

[4.3 编码与单元测试 24](#_Toc68308952)

[4.3.1 用户注册功能 25](#_Toc68308953)

[4.3.2 用户登录功能 25](#_Toc68308954)

[4.3.3 用户密码修改功能 27](#_Toc68308955)

[4.3.4 权限控制功能 27](#_Toc68308956)

[4.3.5 合作方操作功能 28](#_Toc68308957)

[4.3.6 管理员审核功能 29](#_Toc68308958)

[5 集成测试 29](#_Toc68308959)

[5.1为什么要进行软件测试 29](#_Toc68308960)

[5.2 软件测试的步骤 29](#_Toc68308961)

[5.3 集成测试 30](#_Toc68308962)

[5.3.1 用户注册功能测试 32](#_Toc68308963)

[5.3.2 用户登录功能测试 33](#_Toc68308964)

[5.3.3 用户信息管理测试 34](#_Toc68308965)

[5.3.4 权限控制功能测试 35](#_Toc68308966)

[5.3.5 合作方信息管理测试 36](#_Toc68308967)

[5.3.6 管理员审批功能测试 37](#_Toc68308968)

[6 总结与展望 39](#_Toc68308969)

[参考文献 40](#_Toc68308970)

[致谢 41](#_Toc68308971)

摘要

计算机技术发展愈发的蓬勃旺盛，在越来越多的领域和产业上，借助计算机高效的处理能力，计算机网络迅速快捷的信息交互能力与计算机信息方便的传播能力，让原本暮气沉沉的市场泛起了新鲜的活力，刺激市场的同时更是巨大地提高了国民的人均生产力。

但是在当下某些国家地区的单边贸易主义下，国际间的贸易合作关系管理也变成了一个难题，因此，在本科毕设选题上面以“国际企业合作伙伴管理系统的设计与开发”为主题，进行一个国际间企业合作伙伴管理系统的设计与开发，并最终达到一个高可用的状态。

《国际企业合作伙伴管理系统的设计与开发》（后文称“本课题”）将以java为主要的开发语言，使用当下主流的微服务开发模式和DDD（Domain Driver Design）领域驱动设计模型，技术上会使用到一些常见的开源框架或中间件（springboot，jpa，redis等）进行辅助开发，最终结果是开发出一个文档比较完善，接口设计人性化，用户体验良好的基于RESTful接口的国际企业合作伙伴管理系统。开发完成后，用户只需要遵循国际通用的http协议标准，通过调用相应的RESTful api，即可获取相对应的数据。

本课题的设计与开发建立在一个理想的模型下，即各企业均完全在法律的约束下进行公平合作，在这样的基础上，作者将结合相关的经济，市场知识和计算机的高效性，开发一个国际企业合作伙伴管理系统供国内外企业进行参考。此系统基于企业的实际合作管理业务出发，经过软件开发的一系列标准流程，最后实现一个功能较为齐全，用户操作人性化，文档完善的国际企业合作伙伴管理系统。

关键词：合作伙伴管理系统；Java；国际化；RESTful接口开发

Abstract

The development of computer technology is becoming more and more prosperous. In more and more fields and industries, with the help of the efficient processing power of computers, the originally dormant market has become widespread With fresh vitality, while stimulating the market, it also greatly increased the per capita productivity of the people.

However, under the current unilateral trade doctrine in some countries and regions, the management of international trade partnerships has also become a problem. Therefore, the topic of the undergraduate course is "Design and Development of International Business Partner Management System"(hereinafter referred to as "this subject"). As the theme, carry out the design and development of an international enterprise partner management system, and finally achieve a high-availability state.

This subject will use java as the main development language, using the current mainstream microservice development model and DDD domain-driven design model, Technically, some common open-source frameworks (springboot, jpa, redis, etc.) will be used for auxiliary development. The final result is to develop an international enterprise based on RESTful interfaces with complete documentation, humanized interface design, and good user experience. Eventually, the user only needs to follow the http protocol standard and call the RESTful api to obtain the corresponding data.

The design and development of this subject is based on an ideal model, that is, all companies are fully cooperating fairly under legal constraints. On this basis, I will combine relevant economics, market knowledge and computer efficiency to develop An international corporate partner management system. This system is based on the actual cooperation management business of the enterprise.It finally realizes an international enterprise partner management system with more complete functions, user-friendly operation, and complete documentation.

**Keywords:** partner management system; Java; internationalization; RESTful interface development

1. 引言

## 课题研究的背景与意义

随着世界经济与科技水平不断的发展，以及我国以创新为主导的外向型经济和相关政策的大力支持，越来越多的中国企业需要“走出去”，与国外的商业贸易合作伙伴进行深入合作与沟通，为国际友人们提供物美价廉的优质商品或服务，与此同时，越来越多的国外企业也要“走进来”，丰富我国的一些产业链，增加许多就业机会，带动我国经济的进一步发展。在这个过程中，国际上的企业之间必然需要建立更多商业上的合作关系。然而在近年来，西方某些国家的单边贸易主义频频搅局，对我国某些领先的科技企业予以无须有的诬告与制裁，这意味着国际间的合作伙伴的管理方式还亟需改进。

原本二十一世纪应该是一个以合作共赢为主流思想，顺应经济全球化的国际趋势发展的美好时代。我国在三十年代按照既定的发展策略逐步迈向全面小康的社会主义初级阶段，并且提出“2025中国智造”的发展方针。但是以美国为代表的一些国家，在没有充分证据的情况下，滥用国家行政权力，干涉国际贸易，走向单边贸易主义，将多个国家的企业，高校以及个人列入所谓的“实体清单”中，禁止这些实体与美国一些企业进行正常的贸易活动。尽管如此，面临如此危难关头，我国仍然保持一个开放的姿态，稳中求进，抓住时代的机遇，为了让企业的国际化贸易更加高效透明，本课题试图创建一个新型的在线企业合作伙伴管理系统，通过在线管理合作伙伴与签订合同，能够更好的服务于当下这个时代的国际贸易伙伴管理。

## 国内外现状

### 国内主要现状

企业合作方的管理是企业管理中不可忽视的一部分重要内容，在国内，许多企业都有许多的合作方企业，也会建立成熟的合作方管理系统，但是随着时代的发展，出海的需求越来越多，由于不同国家，不同文化之间地政治经济环境各不相同，使用国内地合作管理系统去接入国际贸易伙伴往往会有很多问题出现，例如政策的变动以及货币的汇率变化，所以国际间的贸易合作往往不能简单地直接套用国内贸易合作流程。

### 国外主要现状

在国外，受资本贸易发展时间长的影响，许多大型跨国企业都已经拥有了比较成熟可靠的国际企业合作方管理机制，但是很多步骤还是需要人工手动干预，因此，根据那些已有的成熟机制，并没有有效的结合互联网生态的蓬勃发展，跟上时代发展的潮流。

## 课题研究目的及主要内容

### 课题研究目的

通过设计一种新型的在线企业合作伙伴管理系统，实现在线管理合作伙伴与签订合同，尽力避免单边贸易主义的影响，能够更好的服务于当下这个时代的国际贸易伙伴管理。

### 课题研究的主要内容

在国内本土企业国际化趋势下，对国际贸易伙伴的管理方式需要一些新的变革，基于此目的，通过设计一个简单通用的线上系统来进行国际贸易伙伴的在线管理，并且设计一套成体系的api来作为一个可参考的业务流程标准。

此课题设计的系统在设计之初将可能存在的用户主要分为以下三类：管理员（系统创建者）、合作方（企业）、企业普通注册用户（普通用户）；

(1)系统创建方（管理员）

(2)合作方（企业）

(3)潜在合作方（普通用户）

## 可行性分析

根据软件工程学的标准流程，在一个系统开始设计之前，首先要考虑可行性的问题，主要包含技术可行性，经济可行性，操作可行性三个方面的确定，这些内容涵盖了现有的技术手段是否能够保证课题系统在通用的计算机操作系统上正常运行，课题系统运行在物理主机上的经济成本和开发部署课题系统的时间成本是否合理，以及法律和用户操作性是否可以被正常地快速理解并接受。并且可行性分析的实行通常基于对现有系统的充分调研，只有保证对现有系统充分了解后才能更好的设计出解决往常痛点的优秀系统。

### 技术可行性研究

技术可行性研究的目的是确认当下将要承载课题系统的硬件设备算力是否达标，能够在用户可接受的响应时间内返回正确结果。在经过对市面上成熟的主流技术做了调研后发现，技术已经到达了成熟可用的阶段，另外半导体硬件设备的长足发展也为系统提供了良好的运行基础。系统开发在普通的消费平台进行，开发完成后还会模拟真实线上环境进行测试，因此从技术上来看是可行的。

### 经济可行性研究

从经济可行性上来分析，结合前面对国际企业合作方管理现状的研究，可以判断就当前的国际企业合作方管理还是存在一定的发展滞后性，一个高效可行的线上合作方管理系统可以很好的帮助企业去进行自动化的合作方管理，并且展示合作信息，以便吸引更多的合作方加入进来，藉此带来一定的经济收益。

### 操作可行性研究

就目前的合作方管理流程来说，大多数情况都是重复，繁琐的形式化工作，其中通过计算机进行数字化自动化改造的空间非常巨大，并且经过缜密的法律咨询和调查，系统合法性也有良好的保障，从操作可行性上来说，是没有什么问题的。

所以本系统通过三方面的分析，可以得出该系统有较高的开发可开发性。

## 论文内容安排

论文正文一共由六个模块构成。

第一部分：引言，主要叙述了国内外企业在合作伙伴管理业务上的现状以及相关背景、研究线上国际企业合作伙伴管理系统的意义和该系统研发后对现今企业的有利影响。

第二部分：环境与技术栈简介，这一块主要简单介绍了课题项目开发的系统环境以及用到的一些技术，系统开发工具及相关的服务器运行环境分析，此外还会对论文中使用的计算机语言Java和用到的一系列相关技术栈进行详细的说明。

第三部分：需求分析，通过对现行的国际企业合作伙伴管理系统进行一系列的深度调研，然后总结出相关的业务痛点和需求，并且对这些需求进行细致的划分，罗列了企业合作伙伴之间信息交互，合同签订，款货结佣等基本功能。另外，对于系统的运行效率，运行的可靠性，系统开发成本与后期运维便捷性几个重要方面，本系统也要做到细致的考量。

第四部分：概要设计与详细设计，按照软件工程学的八大基本流程，需求分析清楚后开始进入系统的设计阶段，由于篇幅与内容相关性比较高，这两个流程在第四部分一起说明。这一部分主要的内容是在需求分析总结出来的几个模块的基础上，先是进行简要的流程设计，画出几个核心实现的流程图，简明扼要的总结出大致需要使用到的数据结构；而后在详细设计流程中针对性的扩充细节内容，以ER图的形式补充数据库的设计。正如软件工程课程所学，编码的时间通常只占开发的20%，经过对系统的悉心设计之后，通过计算机语言编码实现也就有了夯实的基础。

第五部分：编码与测试，这一部分的主要内容是将国际企业合作伙伴管理系统的核心实现进行展示与说明，并且建立在软件测试课程所学知识的基础上，设计出详细的测试用力，完成相关接口的单元测试和之后的系统集成测试，尽最大努力减少系统可能出现的bug或者后期运行的不稳定因素。

第六部分：后期运维与总结，对该课题国际企业合作伙伴管理的设计与开发进行总结性叙述，包括对课题研究成果在学术上和应用上的难点突破和亮点进行展示，课题研究过程中使用到的一系列研究方法进行说明，除此之外，对与该系统的不足和优化方向也要进行补充，最后简单发表自己对于国际企业合作伙伴管理前景的展望与期待。

1. 系统开发技术及环境

## 编程语言Java

Java 是迄今为止的第一大编程语言和开发平台。它有助于企业降低成本、缩短开发周期、推动创新以及改善应用服务。如今全球范围内有数以百万计的开发工程师使用Java进行软件开发，Java 仍是企业和开发人员的首选开发平台。

Java作为一款新时代的，面向对象的主流编程语言，同时也是一款基于jvm的半编译半解释型语言，编写的源代码通过java编译器编译为字节码文件后，由jvm进行加载，然后运行在jvm虚拟机上，通过jvm这层中间代理，在各个操作系统平台只需要有对应的可执行文件即可以实现无感知的切换运行环境，因此其兼具开发效率以及运行效率，并且天然具备良好的跨平台性。

Java语言主要分为两大板块，一个是运行环境jre/jvm，以及一个开发工具包jdk。前者为java开发的软件提供一个高效的跨平台运行虚拟机，这个虚拟机是通过c++语言进行实现的，因此Java的运行效率十分高效，开发者不需要为编程语言的运行效率做额外的考虑和优化，只需要将精力关注于业务和逻辑的实现；后者为开发人员提供常见的开发接口和数据结构，诸如网络通信包以及utils工具包，相较于一些早期的高级开发语言，jdk提供的这些api给开发人员带来了极大的便利性，可以很好的提高开发效率。

总而言之，通过使用java作为课题研究系统的开发语言，可以兼顾系统运行效率和兼容性的同时有着高效的开发效率。用户只需要在预计运行课题系统的计算机上安装一个java运行时环境，然后包括课题系统在内的所有基于jvm的应用程序都可以在不超过硬件设备的瓶颈的条件下基于此计算机顺利运行。此外，市场上主流的成熟的操作系统windows、macos以及linux通常会将jre作为基础程序预装其中，因此java程序通常可以做到一次编译，处处运行，开发人员完成对应的功能之后，就可以在各种操作系统中运行，并且基本上不需要在代码层面进行兼容性修改。

## 集成开发环境idea

Idea是一款由jetbrains公司基于java语言开发的商业化集成开发环境，在各个操作系统上都有着对应发行版本，提供了高效统一的开发环境，另外，对于学习或个人使用还提供了community社区版本，相较于商业版本进行了一定的功能缩减，但是仍然保留了像代码智能提示和补全这样的大量的常用功能供用户使用。

Idea同时也是一个优秀的开发平台，它不仅集成了Java的开发环境，而且提供了大量的优质插件和api供用户使用。通过这样的形式，用户可以在插件市场中安装自己所需要的开发工具，也可以使用应用api进行私人化的插件定制。采用Idea平台开发课题系统会具有以下三个优势：

(1)跨平台：在各个主流操作系统上均有发行版，并且页面相对统一。

(2)开发高效：导航栏有各种常用的开发功能，便于用户上手。

(3)开发智能化：Idea提供良好的智能化代码提示功能，极大提高开发效率。

## 数据库MySql

MySQL在过去是一款非常流行，成本低且高效的开源关系型数据库，在被oracle公司收购后分为了商用版本与开源的社区版本，在本课题系统的开发过程中均基于社区版本。关系型数据库基于过去几十年的数据库理论进行一步步迭代成为现今成熟可靠的数据存储仓库，提供了数据设计范式，外键约束和事务等关键的数据设计标准实现，其最大的特点是将所有的数据以表table的形式进行数据存储，并且有着不同的数据库检索引擎，为用户的不同业务不同需求提供多种高效的检索方式。

MySQL-server社区版本在商用版本上做了一定的缩减，减少了部分商业化的后期支持，不过绝大多数数据库的功能均有保留，作为一款标准的关系型数据库，对于SQL查询语言标准有着良好的支持，用户可以通过在不同的编程语言中引入对应的驱动包，然后通过编程语言即可以高效的动态的操作系统所需要的数据。

因此课题系统最终决定采用MySQL数据库，基于成熟的InnoDB检索引擎，可以高效安全的完成多线程读写操作。

## RESTful接口

REST(Representational State Transfer，表现层状态转移)是由Roy T. Fielding [博士在2000年的博士论文](https://link.zhihu.com/?target=https%3A//www.ics.uci.edu/~fielding/pubs/dissertation/top.htm)中提出的，一种面向新时代的，分布式超媒体系统的架构风格。在论文中，Roy T. Fielding博士系统地研究了过去比较成熟的基于网络的应用架构，并且针对性地总结出了一些传统架构的的缺点，比如说首先，随着系统的不断迭代更新，组件的独立性无法得到良好的保障；其次，系统扩展性无法做到非常优秀；最后，对于跨平台跨语言的系统组件开发有着不小的局限性。并且博士为了解决这些传统的设计痛点，首次提出了REST概念，主要用于指导设计Web体系架构的开发。

RESTful接口则是基于REST的设计理念，在http应用层协议的基础上，通过一定的约束和标准，规范数据传输的一系列流程，将web上的所有操作均视为一种资源，以http协议中的URI 作为唯一定位符, 方法体为资源操作限定符，资源数据形式可以任意的数据格式，例如json、xml或者protobuf 等。

本课题系统即预基于RESTful的接口开发出一套规范的数据操作流程，用户可以是各种客户端，像web、android、ios、小程序和操作系统的GUI客户端等都可以通过http协议，根据接口定义的数据规范发送指定的请求，就可以轻松的获取到对应的数据。另外，基于如今Flutter这样的多客户端模板框架出现，结合RESTful接口，可以很轻松的建立其一整套多客户端的管理系统。因此，在阅读相关文献后决定基于这样的一种设计风格开发出一套后台api。

## 应用服务器Tomcat

Tomcat是一款由Apache开源软件基金会开发与管理的轻量级应用服务器，最主要的特点是它不仅在servlet规范基础上实现了一个高效的servlet容器，而且轻量级的静态web页面也可以在Tomcat服务器上进行解析。此外，由于其开源的特性，对于用户访问量不是非常高，另外又受成本限制难以维持一个商业化的付费服务器的中小型应用系统来说再适合不过。

而Servlet是JavaEE规范的一个抽象接口，主要是通过实现它的service()方法来对客户端发起的请求进行业务处理并返回相应的响应。开发人员只需要开发出一个Servlet实现类，并将其与一个url做好映射关系就可以做到动态的对请求进行业务处理。

课题研究的系统最终将被部署在这样的一个应用服务器中，当请求发出后服务器接受到该请求，对该请求的内容进行解析处理后再有服务器返回一个响应对象，最后通过客户端对响应的解析用户就得到了响应结果。

## 系统开发环境Windows10

该课题的开发过程完全在Windows10 2004版本上进行，Windows10是微软开发的一款商业操作系统，桌面端市场占有率长期保持在80%左右，用户基数大，久经市场的考验，许多开发工具与软件都有着良好的适配，基于Windows10操作系统开发可以避免许多兼容性问题并且出现系统问题有着非常完善的解决策略。

本课题国际企业合作伙伴管理系统的开发环境如下：

Operation System：Windows10 2004

Relational Database：MySQL 8.0.19

Key-Value Database：Redis 5.0.9

Web server：Tomcat 9.0.50

Jdk version：JDK 11

Framework：springboot，jpa，redis等

Integrated development environment：idea 2019.3.5

1. 需求分析

根据软件工程的流程指导，经过细致的可行性分析之后，确认了系统实现是确实可行的，下一步则是需要进行需求分析，将用户的需求进行系统化的梳理，并且将需要实现的功能列举出来，以系统功能图的形式划分为各个不同的模块，便于日后的分模块设计，开发以及测试。课题系统需求分析之前还需要对相关的已有项目有一定的了解与熟悉，用于对传统的管理方式进行针对性的痛点分析，这样可以取长补短，做出一个更加完善的系统。

## 系统功能需求

对于一个企业级的管理系统，完善且健全的权限管理模块则是一个不可避免的重要问题，没有完善的权限管理则意味着一个系统可能处处都有漏洞，用户很有可能以一个普通用户的身份去执行一些危险的指令或操作，威胁到系统整体的可用性和数据的安全性，最终导致企业受到经济或名誉上的亏损。

因此，在设计国际企业合作伙伴管理的最开始，就会考虑到对系统权限管理模块需求进行细致的分析。整个系统将划分为两个独立模块，分别为权限管理模块与合作方管理模块。权限管理模块主要实现：多角色多身份的用户鉴权授权机制，具有加密存储的登录注册，动态化的菜单管理功能；合作方管理模块则主要实现合作方业务上的一些具体内容：合作方信息的创建/删除/修改，对应审批流程的数据管理，合作方站点/支付信息的创建/删除/修改等功能。这两个独立模块共同支撑起整个系统的完善功能，给企业带来健全的合作方管理系统。

权限管理模块：

(1)用户注册：实现用户的自动化注册，传入注册所需信息后在数据库中插入新的用户数据，并且为了保证安全性需要对密码进行加密，防止遭受脱库攻击后用户密码泄漏，之后用户可以使用注册的信息来登录系统。

(2)用户登录：实现用户注册后的鉴权和授权功能，用户只需要记录自己的身份信息和密码即可登录系统并自动化授权后根据对应拥有的角色权限开放对应的接口访问，为了与注册信息对应，需要使用注册时使用到的加密算法进行匹配。

(3)用户信息管理：实现用户数据的增加，删除，修改和查询功能，并且对于增删改这几个数据写操作需要进行api的权限控制。

(4)用户密码修改：实现用户密码信息的修改，保障每一个用户的数据安全。

(5)角色信息管理：RBAC权限管理架构中Role角色信息的增删改查，需要管理员权限才能够执行写操作。

(6)权限信息管理：RBAC权限管理架构中Permission权限信息的增删改查，需要管理员权限才能够执行写操作。

(7)菜单信息管理：RBAC权限管理架构引申出来的Menu菜单信息的增删改查，完成用户菜单的动态获取，需要管理员权限才能够执行写操作。

合作方管理模块：

(1)合作方企业信息管理：实现包含合作方地址，银行账户信息以及子站点等合作方信息的创建，申请，查询，修改和删除操作。

(2)合作方合同信息管理：实现合同信息的数据保存，将一些重要的合同信息保存在云端可以有效的解决商业上的一些纠纷。

(3)管理员审批操作：实现系统使用者的审批操作，对每一个合作方都需要进行仔细地人工审核之后才能够将企业正式纳入合作方之中，或者每一份合同都需要管理员审核才能够进入正式生效地阶段。



图1 系统功能图

## 系统性能需求

国际企业合作伙伴管理系统不仅需要进行基础地功能实现，另外系统面向多角色用户开放，性能也是系统是否合格，达到企业级应用的非常重要因素，当系统的响应时间过长或者存在不可用的情况时极有可能会给客户带来一些困扰和麻烦，也会给使用系统的管理方带来经济与名誉上的损失，因此我们需要在实现功能的基础上，保证所有接口能够在一定的时间内返回正确的响应结果，并且并发量达到一个可观的量级。

## 系统运行需求

系统基于x86通用计算机处理器架构进行开发，通过jvm的适配，在各个操作系统上只要满足jvm的运行需求，既可以运行该系统，不过服务器的硬件性能会对系统的处理速度和响应速度带来一定的影响。

开发机器使用的是一台AMD锐龙R5 3500x@4.2GHz处理器，搭配32GB内存，1tb SSD固态硬盘以及100M光纤宽带的主机，在这配置之上的服务器均可以流畅运行。最低配置结合Windows10的要求：

处理器: 1 GHz 或更快的处理器或系统单芯片（SoC）

RAM: 1 GB（32 位）或2 GB（64 位）

硬盘空间: 16 GB（32 位操作系统）或32 GB（64 位操作系统）

服务器: Tomcat9.0.49

## 系统的用例分析

国际企业合作伙伴管理系统主要面向三种角色提供服务：即管理员（系统的创建者）、合作方企业用户和普通用户，对于拥有不同角色身份的用户系统会提供不同的api权限。

系统面向不同角色身份的用户，其提供的服务也各不相同，虽然他们之间的权限可能会有重合部分，但是实际上不同角色身份的用户是有独立的使用规范的。

对于管理员来说，其作为系统的创建者，应该具有系统的全部权限，包含用户信息管理，权限信息管理，合作方信息管理，合同信息管理等功能的全部权限，但是作为“企业管理者”的身份，对于一些功能会有较强的使用倾向，典型的操作就是审核功能，虽然管理员有全部权限，但是像审核这样的功能是可以视作一种主要的管理员操作，还有一些其他的管理员操作如图2所示。



图2 管理员用例图

对于企业合作方而言，作为系统的第一大类用户，需要通过国际企业合作方管理系统与企业管理员进行用户创建，绑定企业，通过审批之后会赋予该用户合作方企业角色的身份，也会获得该角色身份的全部权限，可以对企业信息进行一些管理操作，此外还能够发起和执行一些商业操作例如合作合同申请，具体的用例图如图3所示。



图3 合作方企业用户用例图

对于普通用户，也就是那些在系统中仅注册了的用户，他们可能对企业管理方的一些业务感兴趣，可以称为潜在合作方，对于这些用户，可以将一些不涉及敏感数据的查询操作赋予他们，如图4所示，这样可以让他们能够更好的了解到企业的一些运作数据，进而吸引更多的合作方企业加入进来。



图4 普通用户用例图

1. 国际企业合作伙伴管理系统的设计与开发

## 概要设计

### 系统架构设计



图5 系统架构图

### 数据库设计

在概要设计阶段，我们需要对数据的结构做出明确规范，确定系统数据传输的内容统一性，安全性，并且需要考虑到各种数据库设计的范式，根据这些成熟的设计范式和业务的实际场景，最终设计出一个比较完善的数据库表。

在权限管理模块中，主要使用RBAC(Role Based Access Control, 基于角色的权限控制)机制，其中主要会涉及到三个实体：用户，角色和权限，他们之间分别通过一个中间表来实现多对多的关联关系，为了系统的功能更加完善，在这个基础上还进行了扩展，加入了动态菜单表，用于实现不同角色菜单的动态管理。权限管理相关表如下：

表1 菜单表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| level | int | YES | 0 |
| name | varchar | YES |  |
| parent\_id | int | YES | 0 |
| route | varchar | YES |  |
| sort | int | YES | 0 |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

表2 权限表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| name | varchar | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

表3 角色表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| name | varchar | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

表4 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| eid | varchar | YES |  |
| email | varchar | YES |  |
| id | int | NO |  |
| locale | varchar | YES |  |
| login\_try\_times | int | YES | 0 |
| name | varchar | YES |  |
| password | varchar | YES |  |
| status | int | YES | 0 |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |
| username | varchar | YES |  |

表5 用户角色表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| menu\_id | int | YES |  |
| role\_id | int | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

表6 角色权限表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| permission\_id | int | YES |  |
| role\_id | int | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

表7 用户角色表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| user\_id | int | YES |  |
| role\_id | int | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

接下来进行实际业务实体的设计，业务实体主要包含：合作方企业信息，合作方银行账户信息，合作方子站点信息，审核信息以及合同信息五个部分。通过这些数据实体，可以将一些数据在经过业务逻辑的处理之后进行持久化的存储在硬盘上进行保存。合作方管理相关表：

表8 银行信息表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| account | varchar | YES |  |
| address | varchar | YES |  |
| city | varchar | YES |  |
| country | varchar | YES |  |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| name | varchar | YES |  |
| partner\_id | int | YES |  |
| province | varchar | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

表9 审批记录表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| check\_code | int | YES |  |
| check\_state | int | YES |  |
| check\_time | datetime | YES |  |
| check\_uid | int | YES |  |
| check\_user\_name | varchar | YES |  |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| out\_id | int | YES |  |
| reject\_reason | varchar | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

表10 合同表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| content | varchar | YES |  |
| contract\_no | varchar | YES |  |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| end\_time | datetime | YES |  |
| id | int | NO |  |
| partner\_id | int | YES |  |
| start\_time | datetime | YES |  |
| state | int | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |
| value | decimal | YES |  |

表11 合作方信息表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| bank\_id | int | YES |  |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| email | varchar | YES |  |
| id | int | NO |  |
| market\_scope | varchar | YES |  |
| name | varchar | YES |  |
| payment\_address\_id | int | YES |  |
| phone\_country\_code | varchar | YES |  |
| phone\_number | varchar | YES |  |
| state | int | YES |  |
| uid | int | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

表12 站点信息表

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN\_NAME | DATA\_TYPE | IS\_NULLABLE | COLUMN\_DEFAULT |
| create\_time | datetime | YES | CURRENT\_TIMESTAMP |
| id | int | NO |  |
| partner\_id | int | YES |  |
| site\_url | varchar | YES |  |
| update\_time | datetime | YES | CURRENT\_TIMESTAMP |

## 详细设计

在经过概要设计之后，已经有了整体的逻辑划分，系统被划分为两大主要模块，它们由各个子功能进行实现，但是每个子功能的细节还没有进行严谨的考量与设计，在详细设计阶段，我们可以通过UML统一建模语言或流程图的形式来进行逻辑的梳理，使用UML来进行详细设计还有一个好处是便于不同设计人员的沟通。

### 用户注册功能

用户输入账号密码等关键的信息，用于创建自己的账号，创建前需要校验数据，创建过程中需要注意加密，避免出现明文存储密码的情况，保证用户数据的安全性。



图6 用户注册功能流程图

### 用户登录功能

用户登录功能的实现主要是校验用户身份的合法性，实践中通常会使用身份信息加凭证等方式来实现登录功能，课题系统中使用的是比较成熟可靠的账号密码机制，不过密码存储过程中使用到了比较新颖的Bcrypt加密算法，校验过程中也需要加密后比对。



图7 用户注册功能流程图

### 用户密码修改功能

用户密码修改功能是为了保障用户隐私，企业信息安全设置的功能之一，修改密码需要传入之前的旧密码和将要使用的新密码，后台经过校验后会将旧密码加密后更新在数据库中持久化保存。



图8 用户注册功能流程图

### 权限控制功能

权限控制功能主要基于RBAC的设计架构，用户信息与角色信息绑定，角色信息与权限信息绑定，这样就可以做到灵活的权限控制了。

图9 权限控制功能流程图

### 合作方操作功能

课题系统的合作方作为核心用户之一，其操作也是比较重要的一部分内容，主要包括绑定企业信息，即传入企业的相关信息，后台经过校验后插入到数据库中；签订合作合同，即传入合同相关的信息，诸如开始时间，结束时间，合同金额等，经过校验和管理员审批后即可通过。



图10 合作方操作流程图

### 管理员审核功能

管理员审核功能的实现需要传入合作方信息或合同信息，管理员经过人工审核真实性后触发审批接口或拒绝接口，拒绝需要提供拒绝原因，通过则记录审核人的相关信息即可。



图11 管理员审批流程图

## 编码与单元测试

经过缜密的详细设计之后，对于系统的基本模块有了大致的划分，需要实现的功能也基本确定下来，细致的数据传输实体也得到了确认，接下来需要针对各个功能点进行更为细致的流程梳理，然后整理出相关的逻辑，最后通过Java编程语言实现出来。

权限管理模块主要负责系统的鉴权与授权，保障系统的安全性，使其不会被外界的恶意用户攻击，主要会采用RBAC的设计理念，并对相应的内容做出进一步扩展。

合作方管理模块则是主要负责对应的系统业务功能，用户从普通用户到潜在合作方再到正式合作方，需要进行多个流程的操作并且得到企业管理员的人工审批。此外正式合作方的合同发起也需要得到企业管理员的验证后审批通过。

本课题系统的详细设计基于理想状态设计，作为开发者会竭力做到对真实情况的还原度，以保障系统的可操作性和用户友好性，但是受于个人思维的限制，很多情况还是会有疏漏的地方，需要在后续的维护过程中不断地迭代更新。

### 用户注册功能

用户注册功能的主要流程为：根据传入的注册用户信息，进行数据校验后对密码进行加密后存储至数据库中，并且默认授予访问者权限和普通用户角色。代码展示如下：

public void register(TAuthUser regUser){

TAuthUser byUsername = repository.findByUsername(regUser.getUsername());

if (byUsername != null){

throw new RegisterException("register fail, username has been used.");

}

String originPassword = regUser.getPassword();

String encryptedPassword = AuthUtil.encode(originPassword);

regUser.setPassword(encryptedPassword);

TAuthUser user = repository.save(regUser);

TAuthUserRole visitorRole = new TAuthUserRole();

visitorRole.setRoleId(RegisterRole.VISITOR.getVal());

visitorRole.setUserId(user.getId());

tUserRoleRepository.save(visitorRole);

TAuthUserRole userRole = new TAuthUserRole();

userRole.setRoleId(RegisterRole.USER.getVal());

userRole.setUserId(user.getId());

tUserRoleRepository.save(userRole);

}

### 用户登录功能

用户登录功能则是根据用户传入的账号和密码，在后台读取数据库中的持久化保存数据，校验通过后将权限授予登录用户，这里为了保证系统的性能，引入了redis缓存中间件，保存一些常用的角色信息和权限信息。此外，为了保证用户账号的安全性，还需要进行验证码的校验和登录次数的限制，代码展示如下：

public UserDetails loadUserByUsername(String s) {

UserDetails redisUser = SerializeUtil.string2Obj(stringRedisTemplate.opsForValue().get(s + USER\_DETAILS\_SUFFIX), SystemUser.class);

if (redisUser != null){

return redisUser;

}

TAuthUser authUser = Optional.ofNullable(SerializeUtil.string2Obj(stringRedisTemplate.opsForValue().get(s), TAuthUser.class))

.orElse(tUserRepository.findByUsername(s));

if (authUser == null){

throw new UsernameNotFoundException("user:[" + s +"] doesn't exist.");

}

Set<Integer> roleIds = tUserRoleRepository.findByUserId(authUser.getId())

.stream()

.map(TAuthUserRole::getRoleId)

.collect(Collectors.toSet());

List<TAuthRole> roles = tRoleRepository.findAllById(roleIds);

Specification<TAuthRolePermission> specification =

QueryUtil.buildBatchQuerySpecification("roleId", roleIds);

Set<Integer> permissionIds = tRolePermissionRepository.findAll(specification)

.stream()

.map(TAuthRolePermission::getPermissionId)

.collect(Collectors.toSet());

List<TAuthPermission> permissions = tPermissionRepository.findAllById(permissionIds);

Set<GrantedAuthority> grantedAuthorities = new HashSet<>();

roles.forEach(tAuthRole -> grantedAuthorities.add(new SimpleGrantedAuthority(convertToRole(tAuthRole.getName()))));

permissions.forEach(tAuthPermission -> grantedAuthorities.add(new SimpleGrantedAuthority(convertToPermission(tAuthPermission.getName()))));

SystemUser user = SystemUser.builder().username(authUser.getUsername())

.id(authUser.getId())

.password(authUser.getPassword())

.enabled(authUser.getStatus() == 0)

.accountNonLocked(authUser.getLoginTryTimes() <= 5)

.authorities(grantedAuthorities)

.accountNonExpired(true)

.credentialsNonExpired(true)

.build();

authUser.setLoginTryTimes(authUser.getLoginTryTimes() + 1);

tUserRepository.save(authUser);

stringRedisTemplate.opsForValue().set(s + USER\_DETAILS\_SUFFIX, SerializeUtil.obj2String(user), 30, TimeUnit.MINUTES);

return user;

}

### 用户密码修改功能

用户密码修改，根据传入的旧密码进行校验，校验成功后加密新密码后存储至数据库中。代码如下：

public void changePassword(String oldPassword, String newPassword) {

SystemUser curUser = AuthUtil.getCurUser();

boolean match = AuthUtil.match(oldPassword, curUser.getPassword());

if (!match){

throw new RuntimeException("old password error.");

}

Optional<TAuthUser> user = repository.findById(curUser.getId());

TAuthUser tAuthUser = user.orElseThrow(() ->

new RuntimeException("there is no user named [" + curUser.getUsername() + "]"));

tAuthUser.setPassword(newPassword);

repository.save(tAuthUser);

}

### 权限控制功能

主要基于RBAC的设计架构，然后根据业务情况进行适当的扩展，外加基于拦截器实现的权限控制框架，达到api级别的精密权限控制，以此来保证系统的安全性，如图12所示。



图12 权限控制功能图

### 合作方操作功能

合作方企业绑定操作和申请称为正式合作方企业，绑定合作方企业后用户成为潜在合作方，而申请成为正式合作方企业，则需要经过管理员审批，业务中会通过发送邮件的形式通知管理员尽快进行审核，代码如下：

public TPrmPartner bindWithUser(TPrmPartner tPrmPartner, TPrmBank tPrmBank, TPrmPaymentAddress tPrmPaymentAddress) {

SystemUser curUser = AuthUtil.getCurUser();

Integer userId = curUser.getId();

tPrmPartner.setUid(userId);

tPrmPartner.setState(Constant.PartnerState.POTENTIAL.getState());

tPrmPartner.setBankId(tPrmBank.getId());

tPrmPartner.setPaymentAddressId(tPrmPaymentAddress.getId());

TPrmPartner save = repository.save(tPrmPartner);

tPrmBank.setPartnerId(tPrmPartner.getId());

tBankRepository.save(tPrmBank);

tPrmPaymentAddress.setPartnerId(tPrmPartner.getId());

tPaymentAddressRepository.save(tPrmPaymentAddress);

return save;

}

public TPrmPartner applyToFormalPartner(TPrmPartner tPrmPartner) {

tPrmPartner.setState(Constant.PartnerState.APPLYING.getState());

String[] receivers = emailService.getReceivers();

String text = "尊敬的审核员，你好：\n" +

"\t我方企业[" + tPrmPartner + "]经过充分的考虑决定成为贵企业的正式合作方。\n" +

"\t请贵企业收到邮件后尽快审核并回复。" +

"\tdate:" + new Date() + '\n';

emailService.sendMail(APPLY\_EMAIL\_SUBJECT, text, receivers);

return repository.save(tPrmPartner);

}

### 管理员审核功能

管理员分别要对潜在合作方提出的成为正式合作方的申请进行企业审核，审核通过后才能审批通过，保证企业的真实性，以及对正式合作方发起的合作合同进行审阅后审批通过，保证企业的利益不会受到安全上的威胁，核心代码如下：

public TPrmCheckRecord checkPartner(TPrmCheckRecord tPrmCheckRecord, TPrmPartner tPrmPartner) {

setChecker(tPrmCheckRecord);

tPrmCheckRecord.setCheckCode(Constant.CheckCode.PARTNER.getState());

tPrmPartner.setState(Constant.PartnerState.CHECKED.getState());

tPartnerRepository.save(tPrmPartner);

return repository.save(tPrmCheckRecord);

}

public TPrmCheckRecord checkContract(TPrmCheckRecord tPrmCheckRecord, TPrmContract tPrmContract) {

setChecker(tPrmCheckRecord);

tPrmCheckRecord.setCheckCode(Constant.CheckCode.CONTRACT.getState());

tPrmContract.setState(Constant.ContractState.CHECKED.getState());

tContractRepository.save(tPrmContract);

return repository.save(tPrmCheckRecord);

}

1. 集成测试

## 为什么要进行软件测试

在软件开发的过程中，通常无法做到完全避免所有异常情况，因此，软件测试也就成为了软件系统开发的一个重要组成部分，在测试过程中，我们可以通过科学合理地设置测试用例，采取合适的测试方法，尽力做到对绝大部分异常情况的检测，然后通过不断地迭代去对这些异常情况进行维护。长期以往，系统便在这样地迭代过程中健壮性越来越高。

## 软件测试的步骤

本课题系统的测试将分为两个部分：功能的单元测试以及系统的集成测试，它们分别是由下至上和由上至下的测试流程，对于这两个不同的测试过程会使用到不同的测试方案。

单元测试是在一个软件的最小单位功能完成开发后，为了保证功能的正确性进行的测试工作，这里主要会使用到软件测试课程中学习到的白盒测试方案，将功能进行系统的剖析后设计完善的测试用例，对于有其他功能模块依赖的方法通过Mock来模拟必须的依赖结果，然后在针对设计好的测试用例来进行debug调试。

集成测试是在软件所有功能、模块都开发完成后进行的系统化测试，这一测试阶段是为了验证系统整体的可用性和有效性，许多在单元测试验证之后没有问题的方法，可能会由于依赖的其他模块出现异常导致连锁反应，导致系统出现错误响应甚至崩溃。在集成测试过程中，主要会使用黑盒测试方案，将系统视为一个整体的黑盒，通过postman工具来进行真实请求的模拟，测试系统能否在规定时间内返回正确的响应。

## 集成测试

集成测试主要通过对暴露对外的api进行不同测试用例的访问来进行，对外暴露的api通过文档系统对外展示，整体接口如图13所示。对应的接口细节需要进入文档系统的二级页面中点击具体接口查看，查询到的具体信息如图14所示。

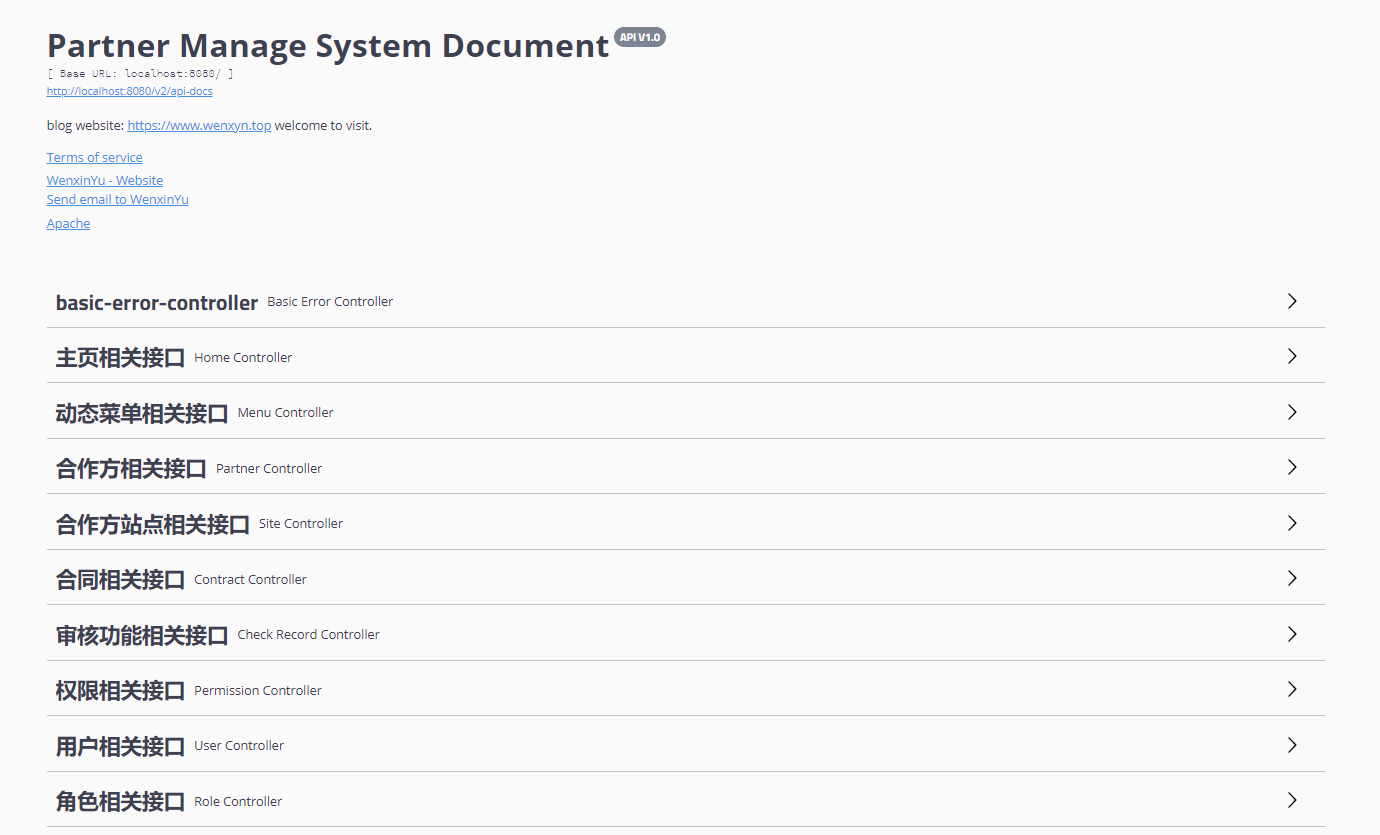


图13 整体接口预览

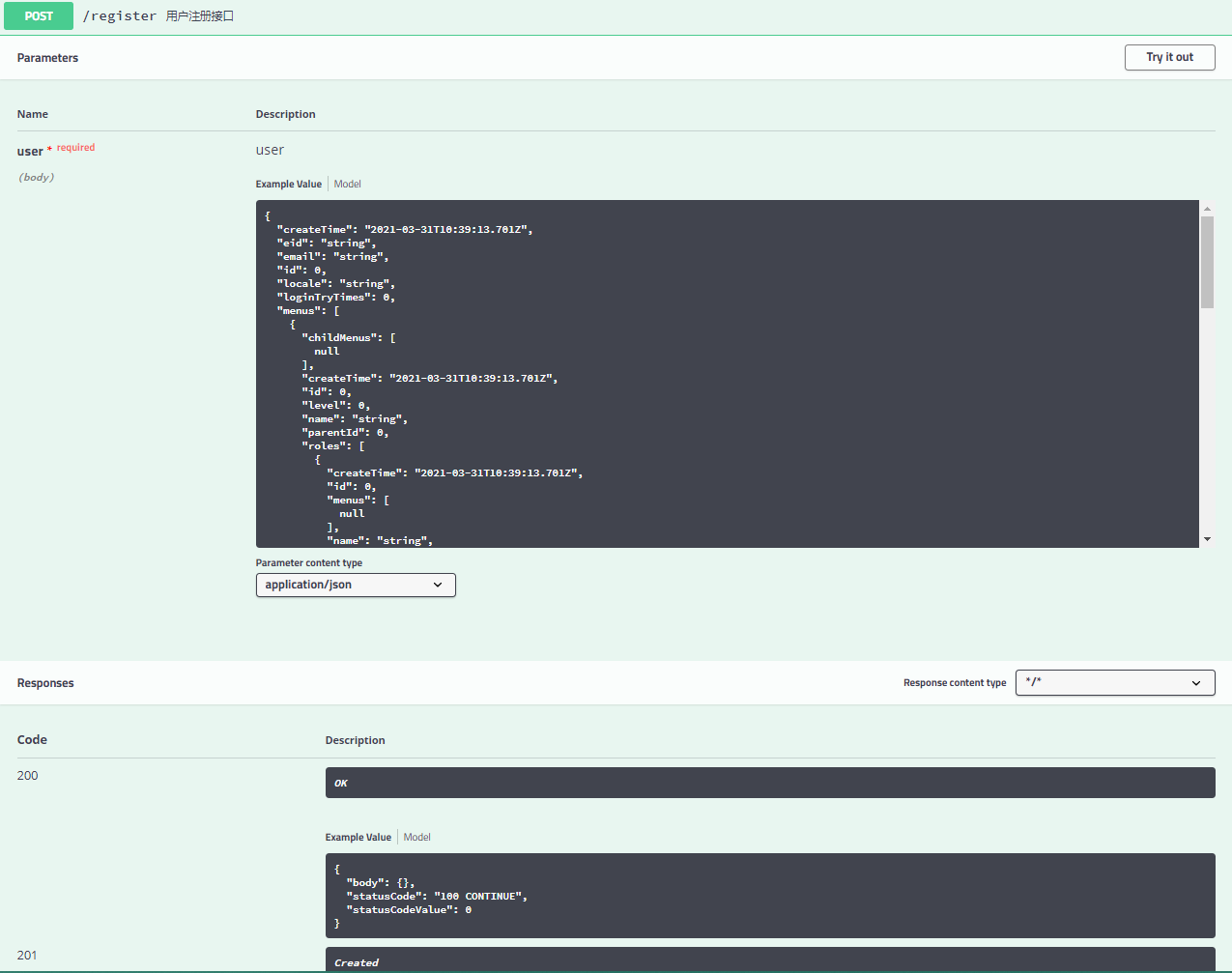


图14 接口细节预览

对于每个接口中涉及的传输对象在文档系统中也有细致说明，每个传输对象的作用，意义，以及对象内部每个字段的含义都有对应的解释，用于和客户端进行对接，前端系统可以根据这些接口的定义传入对应的参数，然后获取正确的返回内容。

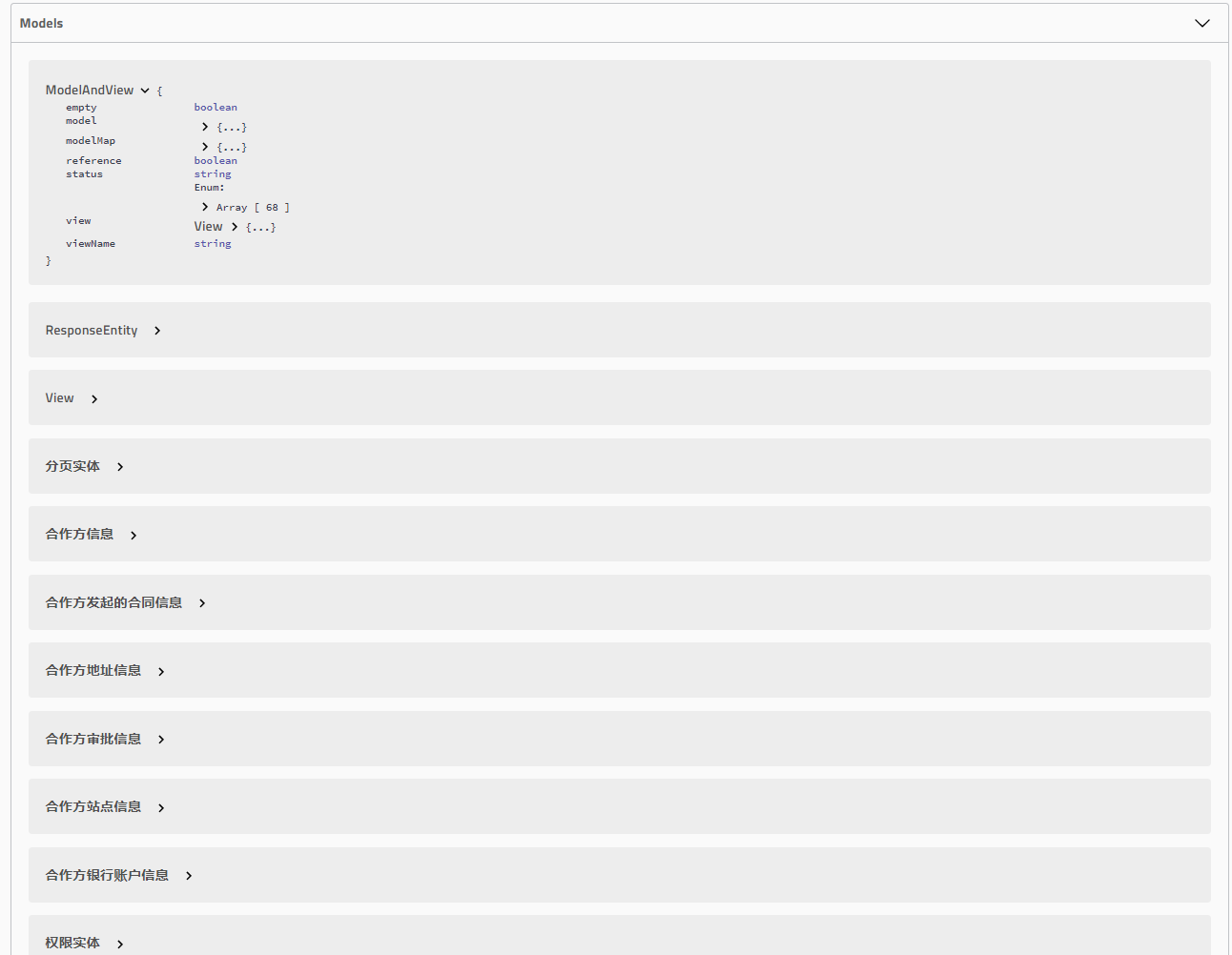


图15 接口实体对象细节预览

### 用户注册功能测试

注册功能的测试通过设计不同的username和password去进行测试。

测试结果：返回的响应内容均正确。

结论：用户注册接口实现成功。

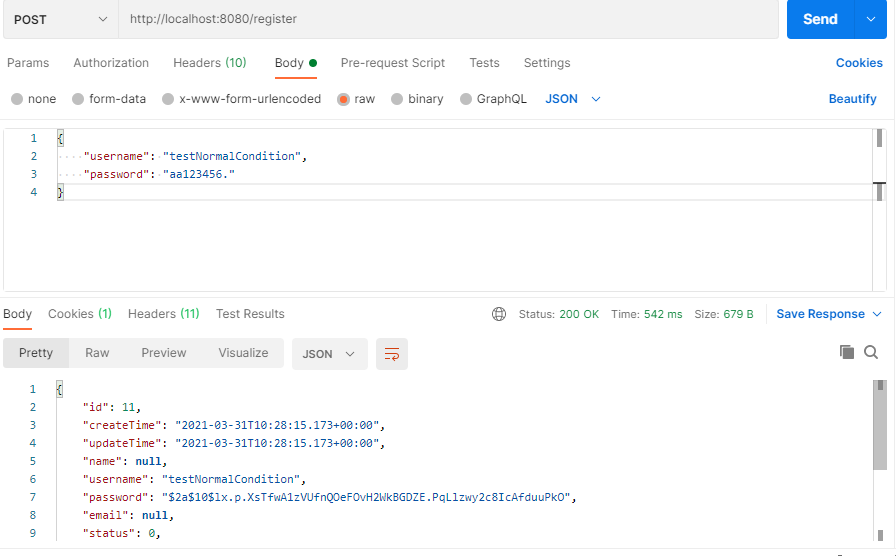


图16 用户注册功能测试

表13 注册功能测试

|  |  |  |
| --- | --- | --- |
| 功能描述 | 注册功能 | |
| 测试数据 | 期望结果 | 实际结果 |
| 用户名：admin 密码：000 | 注册成功 | 注册成功 |
| 用户名：admin 密码：22222 | 注册不成功 | 注册不成功 |
| 用户名：空 密码：34893 | 注册不成功 | 注册不成功 |
| 用户名：admin 密码：空 | 注册不成功 | 注册不成功 |

### 用户登录功能测试

登录功能的测试通过设计不同的username和password去进行测试，期间还需要补充验证码。

测试结果：返回的响应内容均正确。

结论：用户登录接口实现成功。

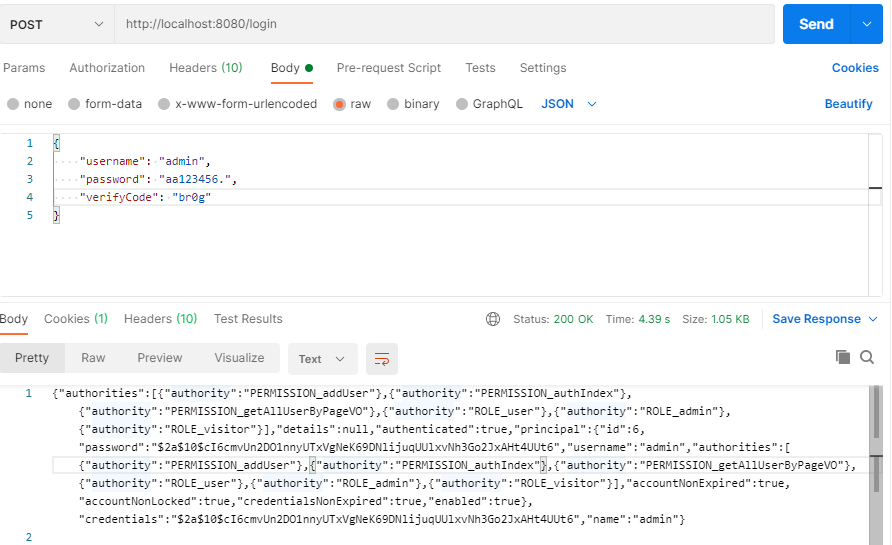


图17 用户登录功能测试

表14 登录功能测试

|  |  |  |
| --- | --- | --- |
| 功能描述 | 登录功能 | |
| 测试数据 | 期望结果 | 实际结果 |
| 用户名：admin 密码：aa123456. | 登录成功 | 登录成功 |
| 用户名：admin 密码：22222 | 登录不成功 | 登录不成功 |
| 用户名：admnni 密码：23345553 | 登录不成功 | 登录不成功 |
| 用户名：空 密码：34893 | 登录不成功 | 登录不成功 |
| 用户名：admin 密码：空 | 登录不成功 | 登录不成功 |

### 用户信息管理测试

用户信息管理的测试主要包含用户信息的增删改查操作，用户密码修改等接口的测试。

测试结果：返回的响应内容均正确。

结论：用户信息管理模块实现成功。

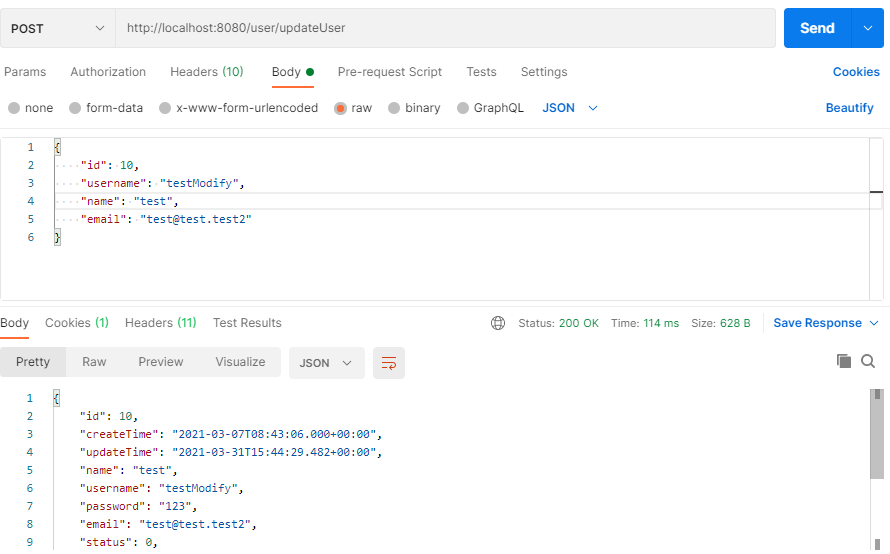


图18 用户信息管理测试

表15 用户信息管理测试

|  |  |  |
| --- | --- | --- |
| 功能描述 | 登录功能 | |
| 测试功能 | 期望结果 | 实际结果 |
| 新增用户 | 成功 | 成功 |
| 修改用户信息 | 成功 | 成功 |
| 删除用户 | 成功 | 成功 |
| 查询用户信息 | 成功 | 成功 |
| 分页查询功能 | 成功 | 成功 |

### 权限控制功能测试

权限控制的测试主要包含权限模块相关信息，包含角色、权限和菜单的增删改查操作等接口的测试。

测试结果：返回的响应内容均正确。

结论：权限控制模块实现成功。

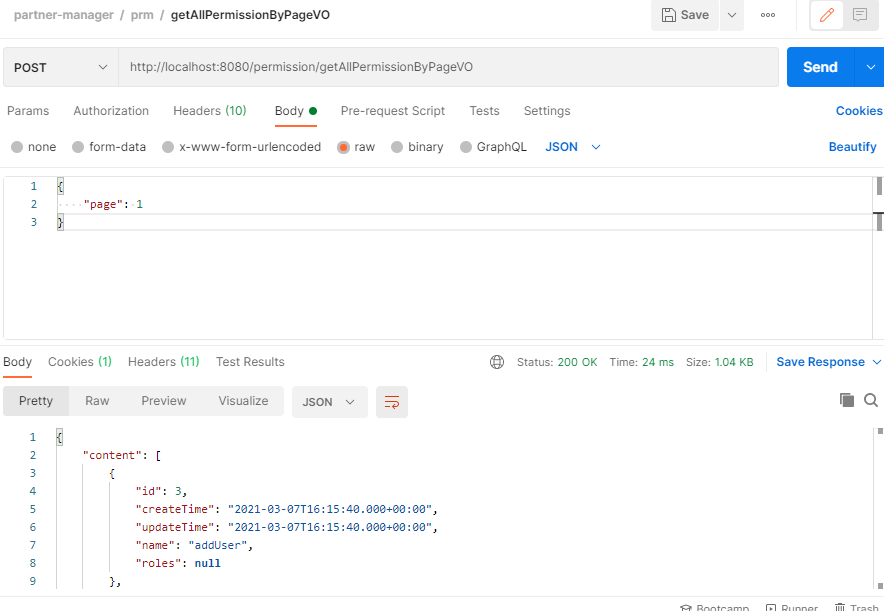


图19 权限控制管理测试

表16 权限控制管理测试

|  |  |  |
| --- | --- | --- |
| 功能描述 | 登录功能 | |
| 测试功能 | 期望结果 | 实际结果 |
| 角色信息管理 | 成功 | 成功 |
| 权限信息管理 | 成功 | 成功 |
| 菜单信息管理 | 成功 | 成功 |
| 分页查询功能 | 成功 | 成功 |

### 合作方信息管理测试

合作方信息管理的测试主要包含合作方相关信息，包含合作方企业绑定、合作方企业申请、合作方银行信息管理，合作方子站点信息管理和合作方合同信息的增删改查操作，用户密码修改等接口的测试。

测试结果：返回的响应内容均正确。

结论：合作方管理模块实现成功。

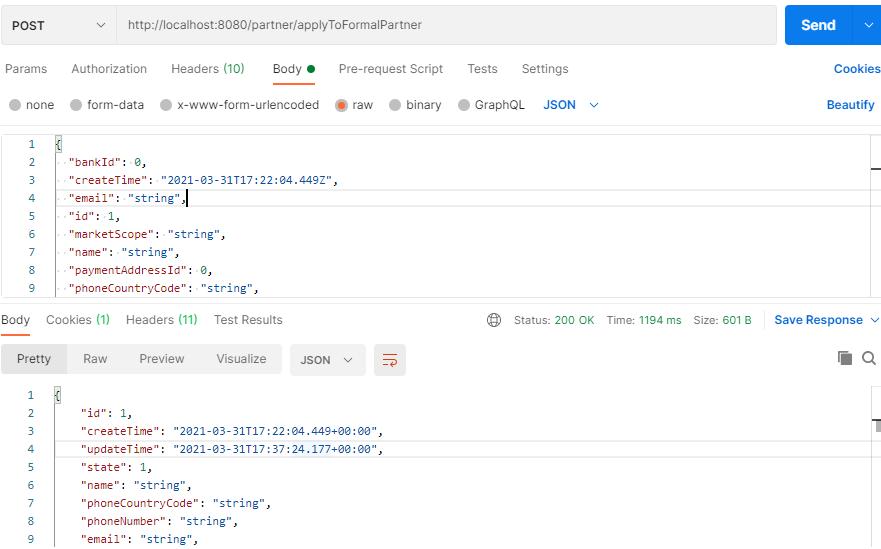


图20 用户信息管理测试

表17 合作方信息管理测试

|  |  |  |
| --- | --- | --- |
| 功能描述 | 登录功能 | |
| 测试功能 | 期望结果 | 实际结果 |
| 合作方企业绑定 | 成功 | 成功 |
| 合作方企业申请 | 成功 | 成功 |
| 合作方合同信息管理 | 成功 | 成功 |
| 合作方银行信息管理 | 成功 | 成功 |
| 合作方站点信息管理 | 成功 | 成功 |
| 分页查询功能 | 成功 | 成功 |

### 管理员审批功能测试

管理员审批功能的测试主要包含管理员的一些核心操作的测试，包括对合作方企业信息的审批操作和对贸易合同信息的审批操作。

测试结果：返回的响应内容均正确。

结论：用户信息管理模块实现成功。

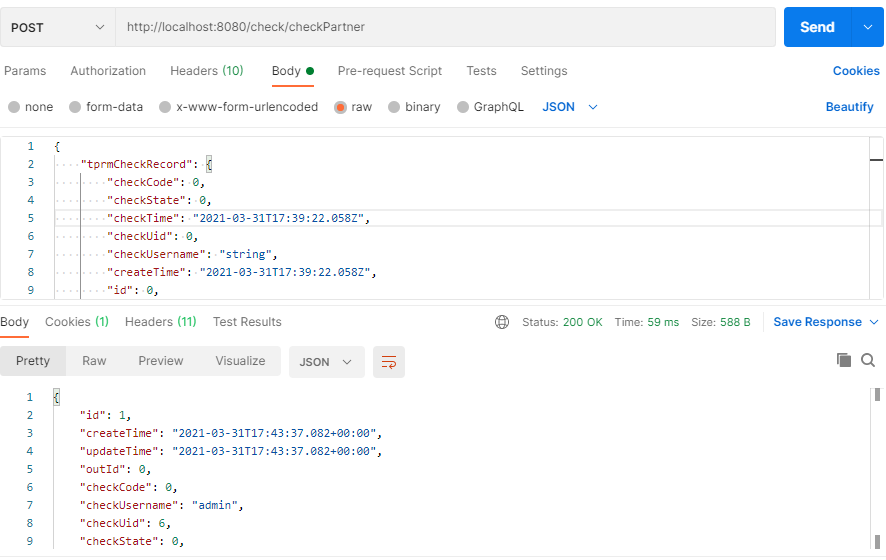


图21 管理员审批功能测试

表18 管理员审批功能测试

|  |  |  |
| --- | --- | --- |
| 功能描述 | 登录功能 | |
| 测试功能 | 期望结果 | 实际结果 |
| 审核合作方企业功能 | 成功 | 成功 |
| 审核合同功能 | 成功 | 成功 |
| 删除合同 | 成功 | 成功 |
| 分页查询功能 | 成功 | 成功 |

1. 总结与展望

经过几个月的课题调研，论文阅读，系统设计与开发，本课题研究的国际合作伙伴管理系统终于初步地完成了既定目标，保证了系统基础功能确实有效的情况下，还保证了服务的可用性，用户在规定的时间内可以得到正确的响应结果，面对将来的国际化趋势，这样的一个系统可以有效的奠定出海基础。

在设计国际合作伙伴管理系统的过程中，将大学求学期间所学到的许多理论知识付诸了成果，实践能力在这一过程中有了长足的进步，系统也在这一过程中不断地调试逐渐完善。不过由于个人对一些实践操作还是略显生疏，对框架代码了解地还不够深入与透彻，在编码实现需求功能地时候还是有许多不算优雅地实现方式，例如RBAC架构中添加的菜单类的实现的不够简练，业务流程中对于现实情况的意外考虑不够全面周到，REST设计风格的不够熟悉等。这意味着系统尚显稚嫩，仍然有着进一步优化的空间。

论文已经快要接近尾声，这是一个学习阶段的系统性结束，但是学海无涯，绝对不会是求学思想的终点站。我对相关业务的知识仍有了解的不足，对与专业基础知识也有可以更进一步的钻研空间，学无止境，将来我也会基于这个课题系统进行不断地完善，竭力让它称为一个真实可用，可以在线上为社会带来价值的企业级系统。

参考文献

1. 蒋莉蓉. 打压持续升级!美商务部将77个实体列入“实体清单”, 涉及中国众多企业､高校及个人[EB/OL]. https://world.huanqiu.com/article/41A2GtLmw7r, 2020. 12. 19. 08:07.
2. Oracle. Oracle Java[EB/OL]. https://www.oracle.com/cn/java/, 2021. 3. 19. 09:13.
3. Fielding, Roy Thomas. Architectural Styles and the Design of Network-based Software Architectures[D]. IRVINE: UNIVERSITY OF CALIFORNIA, 2000.
4. Grimberg. Marcus.REST API & Mobile Application[D]. [Halmstad University](https://webvpn.jxau.edu.cn/http/77726476706e69737468656265737421f5f946d224326d5c770ac7a6804025305a774b/searchFThesis?sw=Halmstad+University&ecode=utf-8&channel=searchFThesis&Field=4): Halmstad University, 2019.
5. 陈霞. 基于RBAC的权限管理系统应用研究[D]. 苏州:苏州大学, 2016.
6. 谢希仁编著. 计算机网络[M]. 北京: 电子工业出版社, 2017. 1: 238-240.
7. 刘庆, 韩蓓蓓主编. 数据库系统概论[M]. 成都: 电子科技大学出版社, 2015.09.
8. 郭鑫. 数据库原理及应用[M]. 重庆: 重庆大学出版社, 2018. 4: 186-251
9. 鲁星, 钱小红, 曾丹主编. 软件工程导论[M]. 武汉: 华中科技大学出版社, 2019. 2.
10. 范晓平主编. 软件工程[M]. 北京: 清华大学出版社, 2019.
11. DJeyaMala, SGeetha著. 马恬煜译. UML面向对象分析与设计[M]. 北京: 清华大学出版社, 2018.06.
12. 朱少民. 软件测试方法和技术[M]. 清华大学出版社, 2010. 7.
13. 韩利凯主编, 高寅生, 袁溪副主编. 软件测试[M]. 北京: 清华大学出版社, 2013. 8.
14. Alok J Saldanha. Java Treeview—extensible visualization of microarray data[J]. Bioinformatics, 2017: 246-248.
15. 罗炜著. 企业合作创新理论研究[M]. 上海:复旦大学出版社, 2002.
16. 岳德震. 一种基于互联网的企业和个人的多元化合作系统[P]. 中国: G06Q10/10;|G06Q50/10, 2020. 3. 20.
17. 孙永福主编. 中国-欧盟中小企业的合作与发展[M]. 北京: 对外经济贸易大学出版社, 2007. 9.

致谢

从课题的确定，到深入了解相关应用的背景，再到了解许多前沿的设计思路，最后成功完成国际合作伙伴管理系统的毕业设计，这期间一步一个脚印，验证了一分耕耘一份收获这样简单的道理，从最开始面对课题的迷茫到最后拨开云雾见日出的豁然，这是人生中一份宝贵的成长历程，期间感谢老师的关注和同学们的帮助，尤其是吴沧海副教授，要是没有她在开题，正文撰写和格式纠正上的耐心指导，可能会需要付出更多事倍功半的努力，甚至会出现南辕北辙的窘境，相信在人生将来的旅途中，这样的经历可以像光一样照亮前方的路。

另外，在论文编写的过程中，阅读、参考和引用了不少著作，这些资料扩展了我的视野，点出了论文的研究方向，给我带来了极大的帮助，这样丰富的科研资料离不开各位科研工作者的辛勤付出，在你们已有的研究基础上，我能够看的更远更广，有了更敏捷的思维。

书山有路勤为径，学海无涯苦作舟。无论明天前路如何，是继续在科研道路上奋勇前行，还是步入社会通过工作给社会带来直接的社会价值，我都会牢记这段难忘的时光。在下一个人生的新起点乘风破浪，创造更大的辉煌。