



東華理工大學
EAST CHINA UNIVERSITY OF TECHNOLOGY

本 科 生 毕 业 设 计 (论 文)

论 文 题 目 :	供应链金融平台贷前授信评级子系统的设计与实现
姓 名 :	匡萃侠
学 号 :	201520180901
班 级 :	1521822Z 班
年 级 :	2015 级
专 业 :	软件工程
学 院 :	软件学院
指 导 教 师 :	张军 (副教授)
完 成 时 间 :	2019 年 5 月 04 日

作者声明

本人以信誉郑重声明：所呈交的学位毕业设计（论文），是本人在指导教师指导下由本人独立撰写完成的，没有剽窃、抄袭、造假等违反道德、学术规范和其他侵权行为。文中引用他人的文献、数据、图件、资料均已明确标注出，不包含他人成果及为获得东华理工大学或其他教育机构的学位或证书而使用过的材料。对本设计（论文）的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本毕业设计（论文）引起的法律结果完全由本人承担。

本毕业设计（论文）成果归东华理工大学所有。

特此声明。

毕业设计（论文）作者（签字）：

签字日期： 年 月 日

本人声明：该学位论文是本人指导学生完成的研究成果，已经审阅过论文的全部内容，并能够保证题目、关键词、摘要部分中英文内容的一致性和准确性。

学位论文指导教师签名：

年 月 日

供应链金融平台贷前授信评级子系统的设计与实现

匡萃侠

Design and Implementation of Pre-loan Credit Rating Subsystem of
Supply Chain Financial Platform

Cuixia Kuang

2019 年 5 月 04 日

摘 要

在我国，随着社会化生产方式的不断深入，中小企业在国民经济中一直发挥着大有可观的作用。然而，融资难、融资贵已经成为限制中小企业发展的最大瓶颈。于是“供应链融资”系列金融产品应运而生。

本文尝试设计并实现了一个以供应链金融为基础的贷前授信评级系统，完成了从线下繁杂、耗时的风险评估模式到利用互联网的线上简约、快速的风险评估模式的转变。该系统将从供应链融资风险来源和融资过程不同阶段分析风险成因以及影响因素，为一些银行以及金融机构提供方便快捷的线上贷前风险评估平台。银行以及金融机构用户通过在系统中录入中小企业的信息以及模型信息，即可快速得到系统给出的建议融资费率等信息，以作为实际融资参考。

本文实现的系统依托于现代互联网公司广泛使用的 MVVM 模式的设计思路，服务器端使用 Spring + Spring MVC + MyBatis 框架编写代码，前端使用 AngularJS 框架进行开发。为供应链金融融资过程贷前阶段提供了确切可行的新型线上风险评估模式以及最终的融资费率参考。

关键字：供应链金融；风险控制； Spring MVC； AngularJS

ABSTRACT

In China, with the continuous deepening of socialized production methods, SMEs have always played a considerable role in the national economy. However, difficulty and high cost of financing have become the biggest bottleneck restricting the development of SMEs. So the "Supply Chain Financing" series of financial products came into being.

This paper attempts to design and implement a pre-loan credit rating system based on supply chain finance, and completes the transition from a complicated and time-consuming risk assessment model to an online simple and rapid risk assessment model using the Internet. The system will analyze risk causes and influencing factors from different stages of supply chain financing risk sources and financing processes, and provide convenient and fast online pre-lending risk assessment platform for some banks and financial institutions. Banks and financial institution users can quickly obtain the recommended financing rate and other information given by the system by entering the information of the SMEs and the model information in the system, as a reference for actual financing.

The system implemented in this paper relies on the design ideas of the MVVM model which is widely used by modern Internet companies. The server side uses the Spring + Spring MVC + MyBatis framework to write code, and the front-end is developed by AngularJS. It provides a new and feasible online risk assessment model and a reference of the final financing rate for the pre-loan stage of the supply chain financial financing process.

Keywords: Supply Chain Finance; Risk Control; Spring MVC; AngularJS

目 录

1 绪论.....	1
1.1 研究背景.....	1
1.2 研究意义及目的.....	1
1.3 研究内容.....	2
1.4 论文结构.....	2
2 系统开发技术基础.....	3
2.1 Java 8.....	3
2.2 Spring.....	3
2.3 Spring MVC.....	4
2.4 MyBatis.....	5
2.5 AngularJS.....	6
2.6 MySQL.....	6
2.7 开发环境.....	7
3 系统分析.....	8
3.1 可行性分析.....	8
3.2 需求分析.....	8
3.2.2 功能结构图.....	8
3.2.3 功能需求.....	9
3.3 数据流分析.....	10
4 系统概要设计.....	12
4.1 系统前端原型设计.....	12
4.1.1 Axure RP 简介.....	12
4.1.2 原型设计.....	12
4.2 系统后端架构设计.....	16
4.3 数据库表关系设计.....	18
4.3.1 数据库表设计.....	18
4.3.2 表设计说明.....	24
5 系统功能的详细设计与实现.....	26
5.1 数据录入模块的实现.....	27
5.2 建模录入模块的实现.....	31
5.3 企业评级模块的实现.....	35
5.4 授信审批模块的实现.....	37
结论.....	40
致 谢.....	41
参考文献.....	42

1 绪论

1.1 研究背景

长期以来，中小企业在国民经济中发挥着举足轻重的作用。据相关数据显示，截至 2015 年末，全国工商登记中，中小企业总数超过 2000 万家，提供了 80% 以上的城镇就业岗位，并且政府的税收总额超过一半由中小企业上缴。由此可见，国民经济的稳健发展与中小企业的繁荣稳定密切相关。但中小企业普遍存在规模小、操作不规范、财务不透明等问题。其中，融资难是阻碍中小企业发展的最大瓶颈。

在这样的背景下，供应链金融逐步发展起来。相较于传统银行信贷，其关注点不再只局限于对单个企业的评价，而更多地关注整个供应链。通过合理处理核心企业及其上下游间的关系，将核心企业的信用输入整个供应链，使中小企业能够从银行得到信贷支持。在我国，自深圳发展银行首次提出供应链金融的概念，各商业银行、物流企业等相继推出相关的融资模式。

而由于供应链金融涉及链条环节长、参与主体众多、环节间彼此影响等问题，造成银行开展此类业务时风险较高。因此，此类业务的开展受到十分严重的制约。2017 年 2 月，在中国小额贷款公司协会举办的第一届会员代表大会第二次会议上，银监会普惠金融部主任李均锋指出，银监会正在研究网络小额贷款的相关指导意见，并建议各地在全国性指导意见和办法出台前能够慎重批设。由此可见，风险是阻碍供应链金融成功实施的关键因素。

为了更好地评估供应链金融的风险，国内外学者做出了许多尝试，通过选取不同的指标，构建不同的模型，来研究供应链金融的风险管理问题。但主要研究多集中于传统的线下融资 1.0 模式，而在互联网融资 2.0 模式和物联网融资 3.0 模式下的供应链金融风险管理研究较少。如何顺应发展趋势，构建在这两种模式下供应链金融的风险控制模型，降低企业的违约概率，也是供应链金融亟需解决的重要课题。

1.2 研究意义及目的

在我国，供应链金融源于 1999 年原深圳发展银行个别分行在当地开展业务时进行的探索与尝试。根据过去十年的数据显示，供应链金融是解决中小企业融资难题的一种有效手段。但与此同时，供应链金融发展得并不稳定，许多银行设立的项目均是通过自我摸索而来，缺乏对业务风险控制、成本规划等方面的认识。由此看出，对供应链金融风险管理的研究是有必要且有意义的。

截至目前，虽然国内外有关供应链金融风险管理的研究正在大量出现，但是其中多数以信贷体系作为基础，以互联网金融为背景的研究较少。同时目前存在的大多数研究是对供应链金融的风险管理进行定性的分析，而对供应链金融的风险进行定量分析的研究并不多见。因此，我决定在前人的基础上，设计出一个供应链风险控制的贷前授信平台，从小微企业数据的录入，再将这些数据代入相应的风控模型，依据对应的公式计算出企业的风险评级，最后给出参考借贷额度以及费率信息。即一些银行，金融机构可依此平台通过建立评价标准模型根据企业各方面数据给予企业合适的借贷额度以及费率，从而达到降低企业违约概率的目的。这对增加银行、金融机构稳定度，促进我国经济又好又快发展有着重要的理论意义。

1.3 研究内容

本文的研究内容是设计并实现一个以供应链金融为基础的贷前授信评级系统，整个项目使用 Maven 构建管理。本文将对此系统实现过程中所涉及到的技术做简要介绍；对供应链金融和风险控制的概念及其内涵进行阐述；从供应链融资风险来源和融资过程不同阶段分析风险成因以及影响因素，根据分析结果设计相应的解决方案并画出 Web 原型页面；以及系统主流程主要功能的代码实现。

1.4 论文结构

本论文的结构大致如下：

第一部分主要介绍的是论文的研究背景、意义、目的以及研究内容；

第二部分将对此系统实现过程中所涉及到的技术做简要介绍，阐明该技术的特点以及在系统实现过程中的具体应用，主要包含 Java 8、Spring、Spring MVC、MyBatis、AngularJS 以及 MySQL，另外还会给出本系统的开发环境简介；

第三部分则开始对系统进行可行性分析、功能需求分析以及数据流分析；

第四部分将对系统进行概要设计，主要包含 Web 原型界面设计、前后端的架构设计以及数据库设计。

第五部分是对于系统重要部分的详细设计与实现，其中会对功能模块做详细设计与实现说明，并对整个系统功能流程进行详细分析介绍。

2 系统开发技术基础

2.1 Java 8

近十年来 Java 都是极为流行的编程语言，同时也积累下了强大的生态系统，我想这也是为什么现在越来越多的企业倾向于选择 Java 开发的原因所在吧。而且 Java 还具备以下优点：

（1）平台无关性：只要平台安装了对应的 Java 虚拟机，那么 Java 就可以在该平台上运行。

（2）纯面向对象：Java 程序是用类来组织的，而类在一个面向对象的系统中，承担的是数据和操作数据的方法的集合。

（3）分布性：Java 提供了很多内置的类库，大大简化了开发人员的程序设计工作，也缩短了项目的开发时间。

（4）安全性：Java 语言经常被用于网络环境中，为了增强程序的安全性，Java 语言提供了一个可以防止恶意代码攻击的安全机制，使其编写的程序具有很好的健壮性。

（5）简单性：去除掉 C 语言和 C++ 语言中难以理解、容易混淆的特性，Java 语言使得程序更加的严谨和简洁，且其还提供了对 Web 应用开发的支持。

正是基于以上 Java 的种种优点，本系统采用 Java 8 编写后台代码，Java 8 是 Java 编程语言开发的一个主要特性版本。它的初始版本于 2014 年 3 月 18 日发布。随着 Java 8 的发布，Java 提供了函数式编程，新的 JavaScript 引擎，用于日期时间操作的新 API，新的流 API 等的支持。

本系统中关于日期和时间的操作都用上了日期时间操作的新 API，不仅线程安全，而且使得整个系统性能更好，代码更简洁；并且在处理 List、Map 集合以及高级查询返回的代码中，为了提高效率，大量使用了 Java 8 提供的 Stream 流的新特性，使集合数据的分组、筛选、排序、重构的效率得到提高。

2.2 Spring

Spring 是一个轻量级框架。Spring 使用的是基本的 JavaBean 来完成以前只可能由 EJB 完成的事情。然而，Spring 的用途不仅仅限于服务器端的开发。从简单性、可测试性和松耦合性角度而言，绝大部分 Java 应用都可以从 Spring 中受益。目的：解决企业应用开发的复杂性；功能：使用基本的 JavaBean 代替 EJB，并提供了更多的企业应用功能；范围：任何 Java 应用。

Spring 是一个轻量级控制反转(IOC)和面向切面(AOP)的容器框架。本系统中所运用

的 Spring 最核心的部分是依赖注入（Dependency Injection），在控制反转（Inversion of Control, IoC）的统一下而实现，该模块被包含在 Spring 核心模块（Spring Core）。对象的构建如果依赖非常多的对象，且层次很深，外层在构造对象时很麻烦且不一定知道如何构建这么多层次的对象。IoC 帮我们管理对象的创建，只需要在配置文件里指定如何构建，每一个对象的配置文件都在类编写的时候指定了，所以最外层对象不需要关心深层次对象如何创建的，前人都写好了。本系统使用 Spring，把 Web 三层架构的核心类交给 Spring 管理，由 Spring 容器根据依赖注入的配置进行初始化不同的业务类，并动态注入相应的依赖属性，这样给本项目的开发带来了很大的便利。

2.3 Spring MVC

Spring MVC 是 Spring 框架的一个模块，Spring MVC 和 Spring 无需通过中间整合层进行整合，拥有控制器，作用跟 Struts 类似，接收外部请求，解析参数传给服务层。MVC 是一种设计模式，即 Model（模型）-View（视图）-Controller（控制器）。

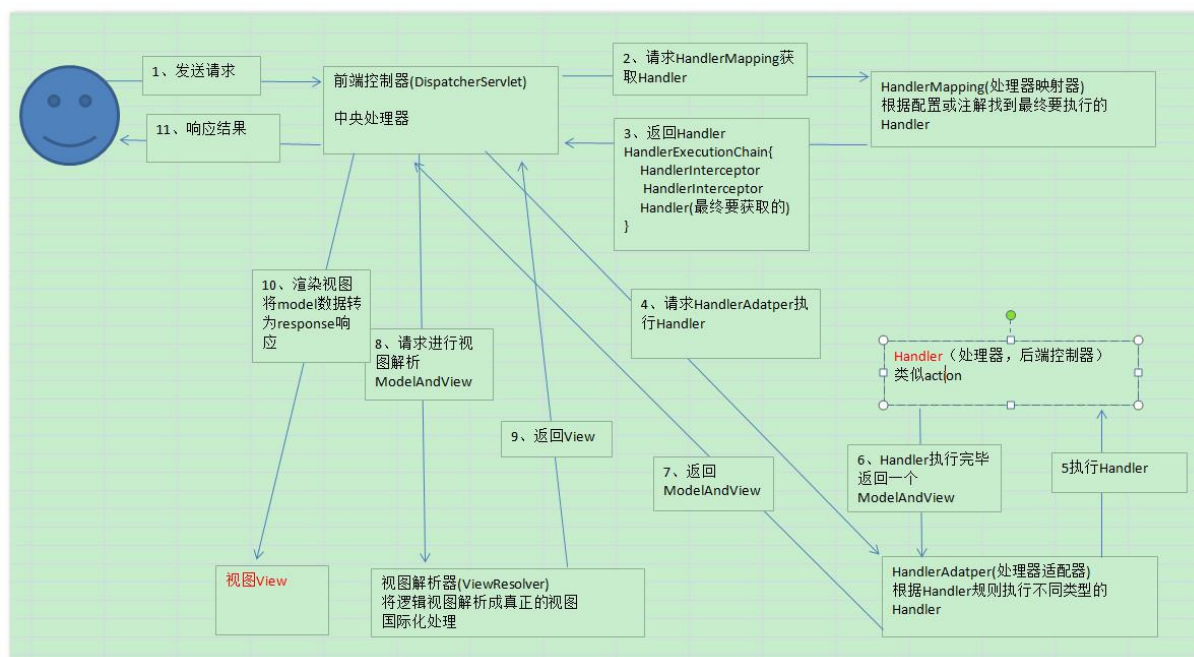


图 2-1 Spring MVC 工作原理图

Spring MVC 提供了一个 DispatcherServlet，作为前端控制器来分派请求，同时，提供了灵活的配置处理程序映射、视图解析、语言环境和主题解析，并支持文件上传。

本系统使用 Spring MVC 基于 Java 的以请求为驱动类型的轻量级 Web 框架，将 Web 层进行解耦，即使用“请求-响应”模型，从工程结构上实现良好的分层，区分职责，简化了 Web 开发。借助于注解，Spring MVC 提供了几乎是 POJO 的开发模式，使得控制器的开发和测试更加简单。这些控制器一般不直接处理请求，而是将其委托给 Spring 上下文中的其他 bean，通过 Spring 的依赖注入功能，这些 bean 被注入到控制器中。

2.4 MyBatis

MyBatis 是一款优秀的持久层框架，它支持定制化 SQL、存储过程以及高级映射。MyBatis 避免了几乎所有的 JDBC 代码和手动设置参数以及获取结果集。MyBatis 可以使用简单的 XML 或注解来配置和映射原生类型、接口和 Java 的 POJO（Plain Old Java Objects，普通老式 Java 对象）为数据库中的记录。MyBatis 的功能架构分为三层：

（1）API 接口层：提供给外部使用的接口 API，开发人员通过这些本地 API 来操纵数据库。接口层一接收到调用请求就会调用数据处理层来完成具体的数据处理。

（2）数据处理层：负责具体的 SQL 查找、SQL 解析、SQL 执行和执行结果映射处理等。它主要的目的是根据调用的请求完成一次数据库操作。

（3）基础支撑层：负责最基础的功能支撑，包括连接管理、事务管理、配置加载和缓存处理，这些都是共用的东西，将他们抽取出来作为最基础的组件。为上层的数据处理层提供最基础的支撑。

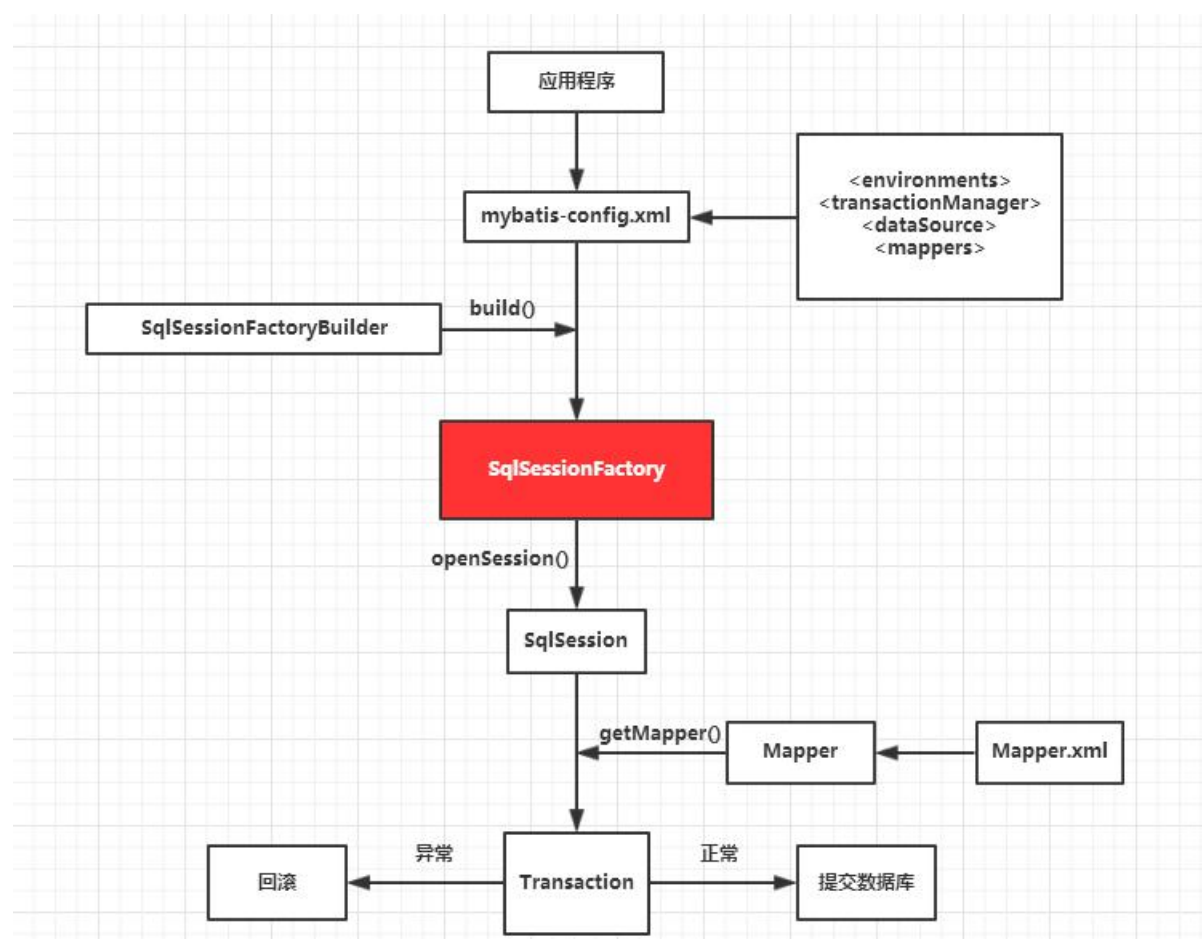


图 2-2 MyBatis 工作原理图

本系统的后台服务器采用 MyBatis 作为持久层框架，MyBatis 将 Dao 接口和 XML 文件里的 SQL 语句建立关系，MyBatis 在初始化 SqlSessionFactoryBean 的时候，找到

mapperLocations 路径去解析里面所有的 XML 文件。Dao 接口并没有实现类，而是通过 JDK 动态代理，返回了一个 Dao 接口的代理对象，这个代理对象的处理器是 MapperProxy 对象。最后系统后台代码通过@Autowired 注入 Dao 接口的时候，注入的就是这个代理对象，我们调用到 Dao 接口的方法时，则会调用到 MapperProxy 对象的 invoke 方法，从而执行 SQL 语句并实现对数据库的各种操作。

2.5 AngularJS

AngularJS 诞生于 2009 年，由 Misko Hevery 等人创建，后为 Google 所收购。是一款优秀的前端 JavaScript 框架，已经被用于 Google 的多款产品当中。AngularJS 有着诸多特性，最为核心的是：MVVM、模块化、自动化双向数据绑定、语义化标签、依赖注入等等。



图 2-3 MVVM 流程图

AngularJS 是为了克服 HTML 在构建应用上的不足而设计的。AngularJS 使得开发现代的单一页面应用程序（SPAs: Single Page Applications）变得更加容易。

本系统采用了 AngularJS 作为前端框架，使整个项目前端组件化、模块化，通过路由功能在不同的单页应用中进行跳转，同时也减轻了前端 JavaScript 代码量，增加了 JavaScript 的复用。因为 AngularJS 的 directive 的行为太过组件化，过了很久才明白其实我们自己编写 JavaScript 也是组件化的。ViewModel 的思维颠覆了传统的 JavaScript 操作 DOM 的行为，既迎合了 MVC 的思想又能够让 JavaScript 的逻辑更加的清晰。使用了 AngularJS 之后就不需要再为查找 DOM 节点以及 JavaScript 动态生成 DOM 节点不能绑定事件而烦恼，使用 AngularJS 只需要将要绑定的事件写在相应的 DOM 上即可，极大方便了 JavaScript 代码的编写。

2.6 MySQL

MySQL 是一个关系型数据库管理系统，由瑞典 MySQL AB 公司开发，目前属于 Oracle 旗下产品。MySQL 是最流行的关系型数据库管理系统之一，在 Web 应用方面，MySQL 是最好的 RDBMS（Relational Database Management System，关系数据库管理系统）应用软件。

本系统选择 MySQL 作为数据存储服务，将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样增加了速度并提高了灵活性。本系统涉及表和字段相对较多，多处为了实现系统功能而将相关表进行关联，提高了系统的整体性能。

2.7 开发环境

本系统的开发环境机器配置如表 2-1 所示：

表 2-1 机器配置

硬件平台	处理器	Inter Core i5
	RAM	8.00 GB
软件平台	操作系统	Win 10 Pro 64 位
	数据库	MySQL 5.7
	集成开发环境	IntelliJ IDEA 2019.1.1
	测试/运行浏览器	Google Chrome

3 系统分析

3.1 可行性分析

首先,从此系统的需求来看,后端的主要功能是对企业各项信息与模型信息的维护,和对这些信息的审核,即状态的变更,以及对于融资费率信息计算的逻辑。前端则是需要设计出相对人性化的交互界面以及信息展示界面,提升用户体验,尽最大努力让使用者灵活操作此系统。目前市面上于此已经有了完整、成熟且可靠的解决方案。

其次,从开发角度来看,本系统和传统的 Web 应用开发方式不同,抛弃了以往的 JSP (Java Server Pages) 动态网页技术,而是使用了较新的 AngularJS 前端框架,不再需要像 JSP 那样编写繁琐的前端代码,极大方便了 JavaScript 代码的编写,且 Spring MVC 对此提供了很好的支持。后台服务器采用成熟、健壮且稳定的 Java 语言进行代码编写。MyBatis 作为持久层框架,提供了成熟的数据库接入接口。Spring MVC 借助于简洁、高效的注解,提供了几乎是 POJO 的开发模式,使得控制器的开发和测试更加简单。另外,本系统所采用的数据库服务是 MySQL 5.7,应对项目中涉及到的各种程度的增、删、改、查操作以及一些状态变更,无论是从数据库角度还是从服务器端的编码方面,都是可行、稳定且高效的。

最后,从前端展示方面来看,在不考虑各类不同浏览器兼容性问题的情况下,本系统使用的 AngularJS 框架,采用组件化的前端编码方式,能够很优雅的解决系统的各种不同需求,同时也映射着更好的 Web 思想: Web Components,未来迟早要到来,Web Components 是趋势。由此,开发效率和运行的稳定性都得到了很好的保证。

3.2 需求分析

3.2.2 功能结构图

本系统的主要功能包括:企业数据录入/审核、建模录入/审核、企业风险评级、人工授信等。在系统中,为了进行完整的流程测试而预留了三个主要角色,分别是超级管理员(admin)、银行录入员(bankmin01)以及银行审核员(bankmch01)。其中,超级管理员负责整个系统的角色创建、菜单管理以及权限分配,银行录入员负责数据录入、建模录入以及企业风险评级,银行审核员负责数据审核、建模审核以及人工授信。本系统由于是被定制工作于银行或者金融机构的环境下,所以不提供用户注册功能,所有用户均由超级管理员(admin)统一进行创建以及分配菜单和权限。下图展示了本系统基于角色不同的大致功能模块划分:



图 3-1 系统功能结构图

3.2.3 功能需求

(1) 超级管理员：本系统实际运行环境的设定是在银行或者金融机构内部，故本系统不提供用户注册功能模块。所以本系统的用户和角色都由超级管理员创建，并给不同用户分配不同的角色，给不同角色设置不同层级的权限以及分配不同的菜单。所以用户只能由超级管理员来进行创建，录入用户信息，并且在用户忘记密码时，需要由超级管理员重置密码。且超级管理员负责角色的创建，设置角色的初始权限并在有需要时进行权限修改，而后再给不同角色分配指定可见的功能菜单。

(2) 银行录入员：银行录入员在系统中作为一种角色而存在，可以被分配给用户，从而用户就获得该角色的权限以及可见功能菜单项。该角色主要有以下三大权限：

① 数据录入：搜集并录入需融资企业的信息，其中，被加入评级模型中的关键选项的信息一定要录入。权限包括：新增、查看和修改。

② 建模录入：根据实际融资产品从六大维度中选择合适的评分选项并设置相应权重形成评级模型。权限包括：新增、查看和修改。

③ 企业评级：选择指定的企业，根据该企业的信息基于评级模型对每个选项进行评分，并根据评分结果由系统自动给出建议融资费率信息。

(3) 银行审核员：银行审核员同银行录入员一样，也是以角色的身份存在于系统中，主要是进行审核工作以及最后给出授信额度。该角色主要有以下三大权限：

① 数据审核：负责审核录入员录入的数据。权限包括：查看和审核。

② 建模审核：负责审核录入员录入的模型。权限包括：查看和审核。

③ 人工授信：根据企业评级结果基于公司授信政策给出最终授信额度。

3.3 数据流分析

根据需求分析画出本系统的顶层数据流图如下所示：

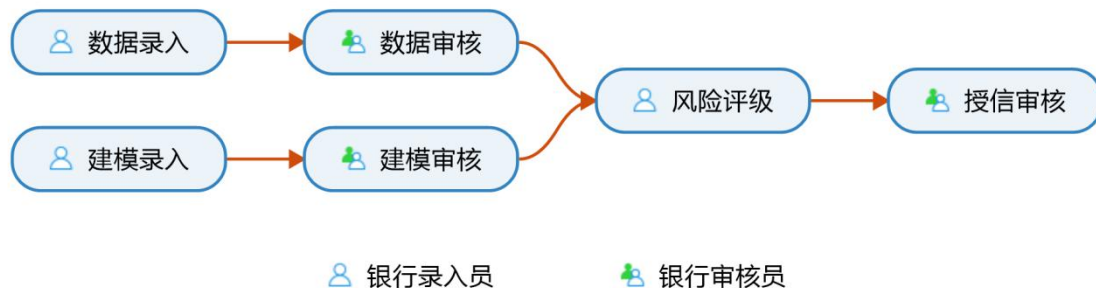


图 3-2 顶层数据流图

如图 3-2 所示，为本系统的顶层数据流图，先由银行或金融机构的录入员录入需融资企业基础信息以及建立相应的评级模型，然后由银行或金融机构的审核员审核录入员录入的企业以及模型信息，审核通过后，录入员才可以对需融资企业进行风险评级，最后再由审核员根据评级结果以及系统给出的融资授信参考信息，并基于公司授信政策给出最终授信额度。这样，一个完整的贷前授信评级流程算是完成了。

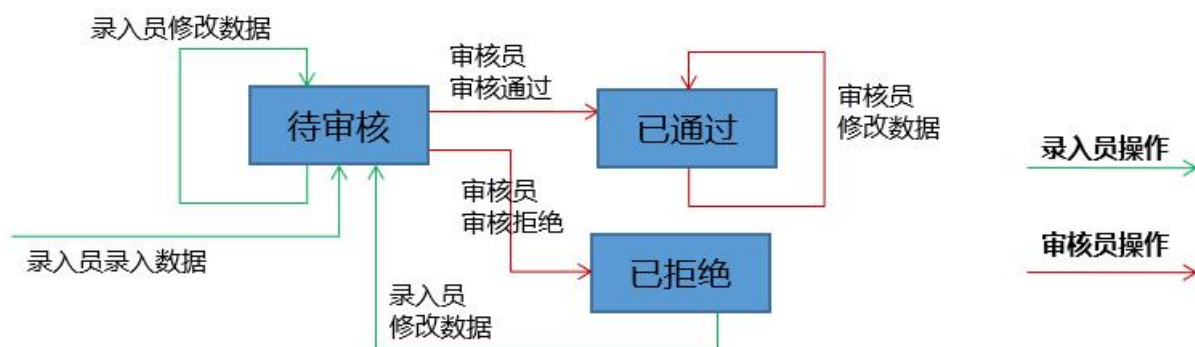


图 3-3 企业信息数据流图

如图 3-3 所示，为企业信息数据流图，银行或金融机构的录入员录入或修改需融资企业信息，录入完后该条记录的初始状态为等待审核。审核员审核等待审核的记录，审核通过后该记录将不能再被修改，只能查看，该记录状态变为审核通过；审核拒绝，该记录状态变为审核拒绝，需录入员再次对企业信息进行修改并提交，成功修改后该记录状态变为等待审核。

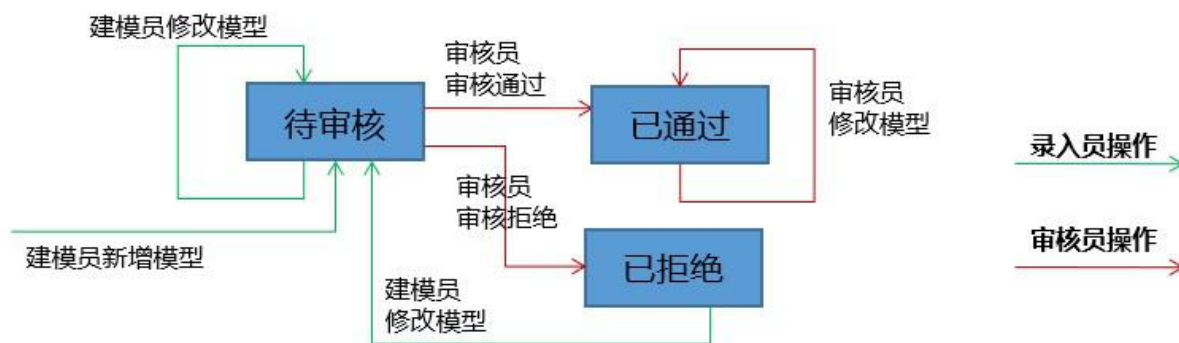


图 3-4 建模信息数据流图


如图 3-4 所示，为建模信息数据流图，银行或金融机构的录入员录入或修改模型信息，录入完后该条记录的初始状态为等待审核。审核过程同企业信息数据流图下的解释说明文字。

4 系统概要设计

4.1 系统前端原型设计

4.1.1 Axure RP 简介

为了简化本系统的开发过程，同时也为了使系统前端界面规范化和美观化，在进行整个系统的代码编写之前，使用 Axure RP（专业的快速原型设计工具）定义需求和规格、设计功能和界面，快速、高效的创建原型，为系统开发指明方向。

Axure RP  是美国 Axure Software Solution 公司旗舰产品，是一个专业的快速原型设计工具，让负责定义需求和规格、设计功能和界面的专家能够快速创建应用软件或 Web 网站的线框图、流程图、原型和规格说明文档。作为专门的原型设计工具，它比一般创建静态原型的工具如 Visio、OmniGraffle、Illustrator、Photoshop、Dreamweaver、Visual Studio、FireWorks 要快速、高效。并且支持 Windows 和苹果 Mac 双系统。Axure RP 能够让您在做出想象中的软件之前，就先体验和使用您的产品！更重要的是，它能帮助我们设计出“正确的解决方案”，没错！是解决方案这个级别的产出，绝不仅仅是简单的页面展示或者是一些炫酷的交互特效的炫耀。这也是为什么全世界 87% 的财富 100 强企业使用 Axure RP 对他们最重要的软件项目进行线框图和原型化的原因。

本系统在进行原型设计时使用的 Axure RP 版本是 2017 年 Axure 公司发布的 Axure RP 8.1 正式版。Axure RP 8.1 正式版新增了许多新特性，例如团队协作功能升级、增加了多个新的流程图原件以及设置元件样式时可以即时预览样式的变化等等，以上新特性也在本次系统进行原型设计的过程中提供了极大的便利。

4.1.2 原型设计

经过初期大量的资料和文献查阅，同时通过对比分析各方学者学术观点，并在对其进行批判分析的基础上试着形成自己的观点，对供应链金融融资过程风险来源和不同阶段的风险成因以及影响因素作出了比较全面的分析总结。

（1）数据录入。

从六大维度来录入数据，每个企业在每个维度都只有一条数据。以企业为单位，按维度划分来通过页面手工录入每个选项。后续将考虑支持批量导入数据的功能。六大维度分别为：企业基本信息、企业财务状况、企业业务往来、企业进销存数据、法人基本信息和主要股东情况。为了提升页面的可复用性，新增页、修改页、查看页以及审核页

将使用相同页面，只是页面属性与实际显示有所不同。下面将展示一些数据录入模块主要页面的原型设计图：

企业名称	注册资本	成立日期	经营期限	办公所在地	状态	操作
神州霞光有限公司	500万	2017-03-23	10年	深圳	等待审核	<button>查看</button> <button>修改</button> <button>审核</button>
贝塔贸易有限公司	200万	2017-03-22	20年	北京	审核通过	<button>查看</button> <button>打印</button>

分页: 上一页 1 下一页 共一页(共一条记录) 到第 页 确定

图 4-1 数据录入列表页原型

如图 4-1 所示，为数据录入的列表页原型，展示了已经录入系统的企业基本信息项，包括企业名称、注册资本、成立日期、经营期限以及办公所在地。在刚新增完一条企业信息后，其初始审核状态为等待审核，该列表页将显示企业数据的实时状态，状态包括三种：等待审核、审核通过以及审核拒绝。同时该页面也提供了新增企业数据（限录入员）功能的入口，对于以录入系统的企业记录，提供查看、修改（限录入员）以及审核（限审核员）功能的入口。

业务往来企业名单

业务往来持续时间(年)

交易总额(万)

应收账款总额(万)

应付账款总额(万)

应收账款周转天数(天)

营业周期(天)

违约次数(次)

法律纠纷(次) 请输入数值

返回列表 上一步 下一步

图 4-2 数据录入编辑页原型

如图 4-2 所示，为数据录入编辑页中企业业务往来选项卡的原型页面，其它选项卡风格与之相同，在后期编码实现阶段将为各个输入框加入正则表达式校验，包括格式校验、输入字符长度校验和必填字段的非空校验，本系统所选用的前端框架 AngularJS 校

验功能的实现提供了良好的支持。

（2）建模。

正是有了前面数据录入模块的基础，建模过程就显得相对简单了。建模的过程就是从六大数据维度中选择某些数据项，并为其赋予权值。但是并非所有数据都能被选入模型，例如企业业务往来名单，于是这类数据或其他一些不能被选入模型的数据就需要授信审核员根据公司政策进行人工评级。下面将展示一些建模录入模块主要页面的原型设计图：



模型编号	模型名称	建模人	建模时间	状态	操作
201703220001	富士康模型	张三	2017-03-23	等待审核	查看 修改 审核 打印
201703220002	包谷网模型	李四	2017-03-22	审核通过	查看 打印

图 4-3 建模录入列表页原型

如图 4-3 所示，为建模录入的列表页原型，展示了已经录入系统的模型基本信息项，包括模型编号、模型名称、建模人以及建模时间，审核状态信息同数据录入建模页的介绍。同时该页面也提供了新增企业数据（限录入员）功能的入口，对于以录入系统的企业记录，提供查看、修改（限录入员）以及审核（限审核员）功能的入口。



业务往来持续时间(年)	交易总额(万)	应收账款总额(万)	应付账款总额(万)
好 3 < 值 ≤ 5 6	好 3 < 值 ≤ 5 6	好 3 < 值 ≤ 5 6	好 3 < 值 ≤ 5 6
中 1 < 值 ≤ 3 4	中 1 < 值 ≤ 3 4	中 1 < 值 ≤ 3 4	中 1 < 值 ≤ 3 4
坏 0 < 值 ≤ 1 1	坏 0 < 值 ≤ 1 1	坏 0 < 值 ≤ 1 1	坏 0 < 值 ≤ 1 1

违约次数(次)	法律纠纷(次)	应收账款周转率	应收账款周转天数
好 3 < 值 ≤ 5 6	好 3 < 值 ≤ 5 6	好 3 < 值 ≤ 5 6	好 3 < 值 ≤ 5 6
中 1 < 值 ≤ 3 4	中 1 < 值 ≤ 3 4	中 1 < 值 ≤ 3 4	中 1 < 值 ≤ 3 4
坏 0 < 值 ≤ 1 1	坏 0 < 值 ≤ 1 1	坏 0 < 值 ≤ 1 1	坏 0 < 值 ≤ 1 1

图 4-4 建模录入编辑页原型

如图 4-4 所示，为建模录入编辑页中企业业务往来选项卡的原型页面，其它选项卡风格与之相同。可根据实际需求将需要入选作为评定标准的数据项勾选上，并填写对应数值即可。



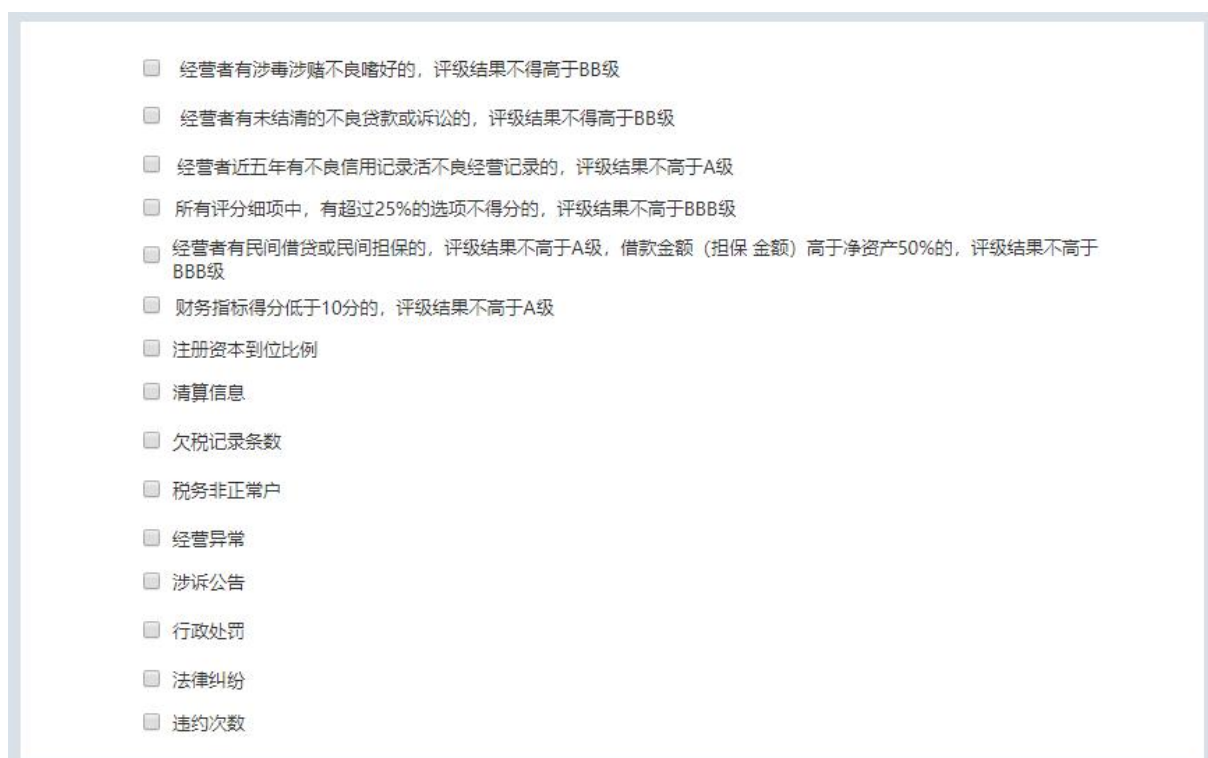
该原型页面用于编辑权值，包含六个输入框，每个输入框右侧都有一个百分号（%）。每个输入框下方都有一行提示文字：“必须输入正整数,且小于一百”。

企业基本信息维度权值	企业财务状况权值
企业业务往来权值	企业进销存数据权值
法人基本信息权值	主要股东情况权值

页面底部有三个按钮：返回列表（蓝色）、上一步（白色）、下一步（白色）。

图 4-5 建模权值编辑页原型

如图 4-5 所示，为建模录入编辑页中为六大维度设置权值的原型页面，六个维度的权值加起来必须等于一百。



该原型页面用于设置特例规则，包含一个列表，每个规则项左侧都有一个复选框（checkbox）。规则项内容如下：

- ☐ 经营者有涉毒涉赌不良嗜好的，评级结果不得高于BB级
- ☐ 经营者有未结清的不良贷款或诉讼的，评级结果不得高于BB级
- ☐ 经营者近五年有不良信用记录活不良经营记录的，评级结果不高于A级
- ☐ 所有评分细项中，有超过25%的选项不得分的，评级结果不高于BBB级
- ☐ 经营者有民间借贷或民间担保的，评级结果不高于A级，借款金额（担保 金额）高于净资产50%的，评级结果不高于BBB级
- ☐ 财务指标得分低于10分的，评级结果不高于A级
- ☐ 注册资本到位比例
- ☐ 清算信息
- ☐ 欠税记录条数
- ☐ 税务非正常户
- ☐ 经营异常
- ☐ 涉诉公告
- ☐ 行政处罚
- ☐ 法律纠纷
- ☐ 违约次数

图 4-6 建模特例规则页原型

如图 4-6 所示，为建模录入编辑页中特里调整规则的原型页面，如果部分指标对客户风险情况判断存在重大影响的，系统也支持对评级进行特别调整的规则设置。最终评

级结果将按照客户得分映射计算，经过特别调整后得出客户的最终信用等级。

信用评级	评分参考值	信用评级系数(%)	建议利率(%)	建议费率(%)	建议期限(天)
A	80 < 值 ≤ 100	100 < 系数 ≤ 100	5	1	360
B	65 < 值 ≤ 80	80 < 系数 ≤ 100	6	1.5	300
C	55 < 值 ≤ 65	60 < 系数 ≤ 80	8	2	180
D	40 < 值 ≤ 55	50 < 系数 ≤ 60	10	2.5	90
E	30 < 值 ≤ 40	30 ≤ 系数 ≤ 50	12	3	60
F	0 ≤ 值 ≤ 30	0	15	3.5	0

基础信用额度	<input type="text"/>	基础信用额度=预估年出货金额 × 帐期天数 ÷ 360
币种	<input type="text" value="请选择"/>	
模型名称	<input type="text"/>	6~20个字母、数字、点、减号或下划线

图 4-7 建模评级划分页原型

如图 4-7 所示，为建模录入编辑页中评级划分的原型页面。加权平均值的计算示例如下：假设该模型 6 个维度的总分是：40，80，70，150，85，75。假设企业 A 在 6 个维度的得分分别是：35，60，25，120，80，65。假设该模型六大维度的权重分别是：5%，10%，15%，20%，25%，25%。则企业 A 在这个模型下的加权平均值计算：

$$(35/40)*100*0.05+(60/80)*100*0.1+(25/70)*100*0.15+(120/150)*100*0.2+(80/85)*100*0.25+(65/75)*100*0.25=64.04$$

评级级别划分示例表如下所示，各信用等级划分、建议额度和建议利率可自定义配置：

表 4-1 评级级别划分示例表

序号	信用等级	得分	建议额度	建议利率
1	AAA	≥80	5000 万	5%
2	AA	≥70	1500 万	6%
3	A	≥60	800 万	8%
4	BBB	≥50	100 万	10%
5	BB	<50	50 万	12%

4.2 系统后端架构设计

本系统的 SpringMVC-AngularJS 技术框架采用主流开源框架搭建，按照分层的开发

架构，自底向上依次为：数据访问层、服务及逻辑层、门面控制层和页面展示层，整个项目使用 Maven（项目管理工具）构建管理。后台服务器使用 Spring MVC 处理 Http 请求，底层使用 MyBatis 框架进行数据的持久化处理，并采用性能稳定、分工明确的 Dao、Service、Manager 和 Controller 四层架构：

（1）Dao（数据访问对象）层：通过 MyBatis，Dao 接口和指定 XML 文件里的 SQL 语句建立联系。Dao 层主要做数据持久层的工作，负责与数据库进行联络的一些任务都封装在此，Dao 层的设计首先是设计 Dao 层的接口，然后在 Spring 的配置文件中定义此接口的实现类，然后就可以再模块中调用此接口来进行数据业务的处理，而不用关心此接口的具体实现类是哪个类，显得结构非常清晰，Dao 层的数据源配置，以及有关数据库连接参数都在 Spring 配置文件中进行配置。

（2）Service（业务）层：Service 层主要负责业务模块的应用逻辑应用设计，同样是首先设计接口，再设计其实现类，接着再 Spring 的配置文件中配置其实现的关联。这样我们就可以在应用中调用 Service 接口来进行业务处理。Service 层的业务实现，具体要调用已经定义的 Dao 层接口，封装 Service 层业务逻辑有利于通用的业务逻辑的独立性和重复利用性。程序显得非常简洁。在 Service 业务层按顺序执行对数据库的数据新增（Insert），数据修改（Update），数据删除（Delete）操作，如果哪一步出现异常（运行时异常），则以上三个操作都将执行 ROLLBACK（事务回滚）。在 MySQL 里事务是一组不可被分割执行的 SQL 语句集合，如果有必要，可以撤销。这样可以确保数据的一致性。ROLLBACK（事物回滚），则事物执行失败，保证了数据安全。

（3）Manager（管理）层：在 Manager 管理层主要进行创建者、创建日期和修改者、修改日期，以及进行日志的记录，例如在新增（修改）企业数据成功后在日志表中新增一条“新增/（修改）企业信息”记录。

（4）Controller（控制）层：Controller 层负责具体的业务模块流程的控制，在此层要调用 Service 层的接口来控制业务流程，控制的配置也同样是在 Spring 的配置文件中，针对具体的业务流程，会有不同的控制器。我们具体的设计过程可以将流程进行抽象归纳，设计出可以重复利用的子单元流程模块。这样不仅使程序结构变得清晰，也大大减少了代码量。在本系统的 Controller 控制层中抛弃了以往的 ModelAndView 写法，而选择了更为方便、简洁的 ResponseEntity 写法。ModelAndView 方式在前端通过 EL 表达式获取值，如\${person}，ResponseEntity 方式在前端通过 data.person 获取 person 对象，通过 data.person.name 获取属性值。

它们之间的关系：Service 层是建立在 DAO 层之上的，建立了 DAO 层后才可以建立 Service 层，Manager 层又在 Service 层之上，而 Manager 层又是在 Controller 层之下的，因而 Manager 层应该既调用 Service 层的接口，又要提供接口给 Controller 层的类来进行调用，Service 层和 Manager 层刚好处于一个中间层的位置。每个模型都有一个 Service 接口，每个接口分别封装各自的业务处理方法。

4.3 数据库表关系设计

4.3.1 数据库表设计

4.3.1.1 数据录入

数据录入模块一共 5 张主表，分别是企业基本信息表、企业财务状况表、企业业务往来表、企业进销存数据表以及企业法人基本信息表，其它因为系统功能需要还会附加一些附表，在这里不多加描述。以下是 5 张主表的字段结构设计：

(1) 企业基本信息表 (tb_enterprise_information)

表 4-2 tb_enterprise_information 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
统一社会信用代码	social_credit_code	varchar(50)	是
办公所在地	office_location	varchar(50)	是
企业名称	enterprise_name	varchar(50)	是
成立日期	building_time	date	是
经营范围	business_scope	varchar(50)	是
经营期限	time_limit	decimal(16,2)	是
注册资本	registered_capital	decimal(16,2)	是
注册资本到位百分比(%)	registered_capital_percentage	decimal(16,2)	是
主要成员	leading_member	varchar(50)	是
最大持股者及出资信息	max_shareholder	bigint(20)	是
变更信息	change_information	bigint(20)	是
股权变更信息	change_equity_information	bigint(20)	是
员工数	employe_number	bigint(20)	是
动产抵押登记信息	chattel_mortgage	bigint(20)	是
股权出质登记信息	pledge_registration_information	bigint(20)	是
公司类型	company_type	varchar(50)	是
商标注册信息	registration_information	varchar(50)	是
知识产权出质登记信息	intellectual_information	varchar(50)	是
分支机构数量	branch_number	decimal(16,2)	是
审批状态	status	varchar(50)	是
创建时间	create_time	date	是
修改时间	update_time	date	是
创建者	create_user_id	bigint(20)	是
修改者	update_user_id	bigint(20)	是
是否删除	is_deleted	int(1)	

(2) 企业财务状况表 (tb_enterprise_finance)

表 4-3 tb_enterprise_finance 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
基础信息表外键	einid	bigint(20)	是
年产值(万)	annual_output	decimal(16,2)	是
前三大客户占比(%)	top_three_accounted	decimal(16,2)	是
前三大客户购销是否分散	top_three_dispersed	varchar(50)	是
前三大客户定价权	top_three_pricing	varchar(50)	是
前三大供应商占(%)	top_three_supplier	decimal(16,2)	是
前三年主营业务收入及营业额比例	top_three_income	decimal(16,2)	是
股权架构	ownership_structure	varchar(50)	是
征信报告	credit_report	varchar(50)	是
法人担保	corporate_guarantee	varchar(50)	是
法人抵押	corporate_mortgage	varchar(50)	是
现金流(万)	cash_flow	decimal(16,2)	是
主要销售条件(付款条件)	main_condition	varchar(50)	是
资产净值(万)	asset_value	decimal(16,2)	是
负债总额(万)	total_liabilities	decimal(16,2)	是
总资产(万)	total_money	decimal(16,2)	是
总资产周转率(%)	total_turnover_rate	decimal(16,2)	是

(3) 企业业务往来表 (tb_enterprise_business_dealings)

表 4-4 tb_enterprise_business_dealings 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
基础信息表外键	einid	bigint(20)	是
业务往来企业名单	current_enterprises_list	varchar(50)	
业务往来持续时间(年)	current_enterprises_time	decimal(16,2)	
交易总额(万)	total_transaction	decimal(16,2)	
应收账款总额(万)	accounts_receivable_money	decimal(16,2)	
应付账款总额(万)	accounts_payable_money	decimal(16,2)	
应收账款周转天数(天)	accounts_receivable_days	decimal(16,2)	
营业周期(年)	business_cycle	decimal(16,2)	
违约次数(次)	breach_contract_times	decimal(16,2)	
法律纠纷(次)	legal_disputes_times	decimal(16,2)	

(4) 企业进销存数据表 (tb_enterprise_invoicing)

表 4-5 tb_enterprise_invoicing 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
基础信息表外键	einid	bigint(20)	是
进货总额(万)	total_purchases	decimal(16,2)	

存货总额(万)	inventory	decimal(16,2)	
销货总额(万)	total_sales	decimal(16,2)	
出库单	outbound_order	varchar(50)	
库存	stock	varchar(50)	
年平均销售金额(万)	average_year_sales_amount	decimal(16,2)	
月平均销售金额(万)	average_month_sales_amount	decimal(16,2)	
商品平均销售天期(天)	average_selling_commodity	decimal(16,2)	
流动资产总额(万)	total_current_assets	decimal(16,2)	
存货周转率(%)	inventory_turnover_rate	decimal(16,2)	
存货周转天数(天)	inventory_turnover_date	decimal(16,2)	

(5) 企业法人基本信息表 (tb_enterprise_legal_person)

表 4-6 tb_enterprise_legal_person 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
基础信息表外键	einid	bigint(20)	是
姓名	law_name	varchar(50)	
手机号	law_telephone	varchar(50)	
籍贯	law_native_place	varchar(50)	
身份证	law_idcard	varchar(50)	
年龄	law_age	varchar(50)	
学历	law_record	varchar(50)	
银行卡数量(张)	law_bank_card_number	decimal(16,2)	
芝麻信用(分)	law_credit_sesame	varchar(50)	
信用卡数量(张)	law_credit_card	decimal(16,2)	
微粒贷额度(万)	law_particle_credit_line	decimal(16,2)	
银行对账单	law_bank_statement	varchar(50)	
资产(万)	law_assets	decimal(16,2)	
行业年限	law_workingtime	varchar(50)	
薪资(万)	law_salary	decimal(16,2)	
上年度电话账单金额(万)	law_last_tel_money	decimal(16,2)	
缴税状况	law_tax_situation	varchar(50)	
保险信息	law_insurance_information	varchar(50)	
对外投资占资产的比例	law_foreign_investment	decimal(16,2)	
缴费状况	law_water_electricity_status	varchar(50)	
电话账单缴费是否正常	law_telephone_payment	varchar(50)	
近六个月银行对账单贷方发生额(万)	law_last_bank_reconciliation	decimal(16,2)	
近三个月银行对账单贷方发生频率(次)	law_last_bank_times	decimal(16,2)	
其他关联信息	law_other	varchar(50)	

4.3.1.2 建模录入

由于建模表字段与数据录入表字段息息相关，因此建模录入模块的数据表主要是根据数据录入模块来设计的，而其自身模块的表主要有 3 张，分别是模型基础信息表、维度权值表以及特例调整规则表。下面将展示上述 3 张表以及和数据录入模块相关的其中一张表的表结构设计：

(1) 建模企业基本信息表 (tb_modeling_enterprise_information)

表 4-7 tb_modeling_enterprise_information 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
模型 id	model_id	bigint(20)	是
业务字段标识码	code	varchar(50)	是
好 最小值	good_min_value	decimal(16,2)	是
好 最大值	good_max_value	decimal(16,2)	是
好 分数	good_fraction	decimal(16,2)	是
中 最小值	secondary_min_value	decimal(16,2)	是
中 最大值	secondary_max_value	decimal(16,2)	是
中 分数	secondary_fraction	decimal(16,2)	是
坏 最小值	bad_min_value	decimal(16,2)	是
坏 最大值	bad_max_value	decimal(16,2)	是
坏 分数	bad_fraction	decimal(16,2)	是
是	yes_fraction	decimal(16,2)	是
否	no_fraction	decimal(16,2)	是
有	have_fraction	decimal(16,2)	是
无	nothing_fraction	decimal(16,2)	是
强	strong_fraction	decimal(16,2)	是
中	medium_fraction	decimal(16,2)	是
弱	weak_fraction	decimal(16,2)	是
自有	own_have_fraction	decimal(16,2)	是
租赁	lease_fraction	decimal(16,2)	是

(2) 模型基础信息表 (tb_base_information)

表 4-8 tb_base_information 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
模型编号	model_id	bigint(20)	是
模型名称	model_name	varchar(50)	是
币种	currency	varchar(50)	是
基础信用额度	base_line_credit	decimal(16,2)	是
A 评分参考值最小值	a_min_scoring_reference	decimal(16,2)	是
A 评分参考值最大值	a_max_scoring_reference	decimal(16,2)	是

A 信用评级系数最小值(%)	a_min_credit_rating_coefficient	decimal(16,2)	是
A 信用评级系数最大值(%)	a_max_credit_rating_coefficient	decimal(16,2)	是
A 建议利率(%)	a_proposed_rate_interest	decimal(16,2)	是
A 建议费率(%)	a_proposed_rate	decimal(16,2)	是
A 建议期限(天)	a_proposed_time_limit	decimal(16,2)	是
B 评分参考值最小值	b_min_scoring_reference	decimal(16,2)	是
B 评分参考值最大值	b_max_scoring_reference	decimal(16,2)	是
B 信用评级系数最小值(%)	b_min_credit_rating_coefficient	decimal(16,2)	是
B 信用评级系数最大值(%)	b_max_credit_rating_coefficient	decimal(16,2)	是
B 建议利率(%)	b_proposed_rate_interest	decimal(16,2)	是
B 建议费率(%)	b_proposed_rate	decimal(16,2)	是
B 建议期限(天)	b_proposed_time_limit	decimal(16,2)	是
C 评分参考值最小值	c_min_scoring_reference	decimal(16,2)	是
C 评分参考值最大值	c_max_scoring_reference	decimal(16,2)	是
C 信用评级系数最小值(%)	c_min_credit_rating_coefficient	decimal(16,2)	是
C 信用评级系数最大值(%)	c_max_credit_rating_coefficient	decimal(16,2)	是
C 建议利率(%)	c_proposed_rate_interest	decimal(16,2)	是
C 建议费率(%)	c_proposed_rate	decimal(16,2)	是
C 建议期限(天)	c_proposed_time_limit	decimal(16,2)	是
D 评分参考值最小值	d_min_scoring_reference	decimal(16,2)	是
D 评分参考值最大值	d_max_scoring_reference	decimal(16,2)	是
D 信用评级系数最小值(%)	d_min_credit_rating_coefficient	decimal(16,2)	是
D 信用评级系数最大值(%)	d_max_credit_rating_coefficient	decimal(16,2)	是
D 建议利率(%)	d_proposed_rate_interest	decimal(16,2)	是
D 建议费率(%)	d_proposed_rate	decimal(16,2)	是
D 建议期限(天)	d_proposed_time_limit	decimal(16,2)	是
E 评分参考值最小值	e_min_scoring_reference	decimal(16,2)	是
E 评分参考值最大值	e_max_scoring_reference	decimal(16,2)	是
E 信用评级系数最小值(%)	e_min_credit_rating_coefficient	decimal(16,2)	是
E 信用评级系数最大值(%)	e_max_credit_rating_coefficient	decimal(16,2)	是
E 建议利率(%)	e_proposed_rate_interest	decimal(16,2)	是
E 建议费率(%)	e_proposed_rate	decimal(16,2)	是
E 建议期限(天)	e_proposed_time_limit	decimal(16,2)	是
F 评分参考值最小值	f_min_scoring_reference	decimal(16,2)	是
F 评分参考值最大值	f_max_scoring_reference	decimal(16,2)	是
F 信用评级系数最小值(%)	f_min_credit_rating_coefficient	decimal(16,2)	是
F 信用评级系数最大值(%)	f_max_credit_rating_coefficient	decimal(16,2)	是
F 建议利率(%)	f_proposed_rate_interest	decimal(16,2)	是
F 建议费率(%)	f_proposed_rate	decimal(16,2)	是
F 建议期限(天)	f_proposed_time_limit	decimal(16,2)	是
审批状态	status	varchar(50)	是
创建时间	create_time	date	是

修改时间	update_time	date	是
创建者	create_user_id	bigint(20)	是
修改者	update_user_id	bigint(20)	是
是否删除	is_deleted	int(1)	

(3) 维度权值表 (tb_dimension)

表 4-9 tb_dimension 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
模型 id	model_id	bigint(20)	是
企业基本信息维度权值	basic_dimension_weight	decimal(16,2)	是
企业财务状况权值	financial_status_weight	decimal(16,2)	是
企业业务往来权值	business_transaction_weight	decimal(16,2)	是
企业进销存数据权值	invoicing_data_weight	decimal(16,2)	是
法人基本信息权值	legal_information_weight	decimal(16,2)	是
主要股东情况权值	shareholder_weights	decimal(16,2)	是

(4) 特例调整规则表 (tb_exception_adjustment_rule)

表 4-10 tb_exception_adjustment_rule 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
模型 id	model_id	bigint(20)	是
code	code	varchar(50)	是

4.3.1.3 企业评级

企业评级表存放的是具体的需融资企业在与指定模型匹配风险评级后的评级结果，将保存详细的细项评分以及建议融资额度和费率信息。以下是该表的结构设计：

企业评级表 (tb_enterprise_rating)

表 4-11 tb_enterprise_rating 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
企业信息表外键	einid	bigint(20)	是
模型 id	model_id	bigint(20)	是
评级结果	rating_results	varchar(50)	是
最后得分	final_score	decimal(16,2)	是
建议额度	proposed_amount	decimal(16,2)	是
建议期限	proposed_period	decimal(16,2)	是
建议利率	proposed_interest_rate	decimal(16,2)	是
建议手续费率	proposed_commission_rate	decimal(16,2)	是
基本信息分项得分	base_score	decimal(16,2)	是
基本信息分项比例	base_proportion	decimal(16,2)	是

财务状况分项得分	finance_score	decimal(16,2)	是
财务状况分项比例	finance_proportion	decimal(16,2)	是
业务往来分项得分	business_score	decimal(16,2)	是
业务往来分项比例	business_proportion	decimal(16,2)	是
进销存情况分项得分	invoicing_score	decimal(16,2)	是
进销存情况分项比例	invoicing_proportion	decimal(16,2)	是
法人情况分项得分	legal_score	decimal(16,2)	是
法人情况分项比例	legal_proportion	decimal(16,2)	是
主要股东情况分项得分	shareholder_score	decimal(16,2)	是
主要股东情况分项比例	shareholder_proportion	decimal(16,2)	是
创建时间	create_time	date	是
修改时间	update_time	date	是
创建者	create_user_id	bigint(20)	是
修改者	update_user_id	bigint(20)	是

4.3.1.4 授信审批

已经评级的需融资企业，风险控制审核人员将根据评级结果基于公司授信政策给出最终授信额度，并存入授信审批表。以下是该表的结构设计：

授信审批表（tb_enterprise_credit）

表 4-12 tb_enterprise_credit 表

逻辑名	物理名	字段类型	可为空
id	id	bigint(20)	否
企业评级表外键	erid	bigint(20)	是
授信额度	credit_line	decimal(16,2)	是
授信利率（%）	credit_rate	decimal(16,2)	是
信息费率（%）	information_rate	decimal(16,2)	是
授信期限（天）	credit_date	decimal(16,2)	是
状态	status	varchar(50)	是
创建时间	create_time	date	是
修改时间	update_time	date	是
创建者	create_user_id	bigint(20)	是
修改者	update_user_id	bigint(20)	是

4.3.2 表设计说明

（1）建模在首页录入评级划分，点击下一步操作时，除了将本页勾选的信息录入数据库，同时将 model_id（模型编号）录入到评级表中，后续的操作都获取该模型编号。

注：模型编号由系统自动生成。

（2）客户在录入特例调整规则时，不仅录入当前页面信息，同时插入从前一页面获取的 `model_id` 到特例调整规则表中。

（3）根据 `model_id` 可以确定唯一的模型，根据 `model_id + code` 的组合可以精确某一个模型的某一个字段。

5 系统功能的详细设计与实现

供应链金融平台贷前授信评级子系统风控评级模型处理流程总括如下所示：

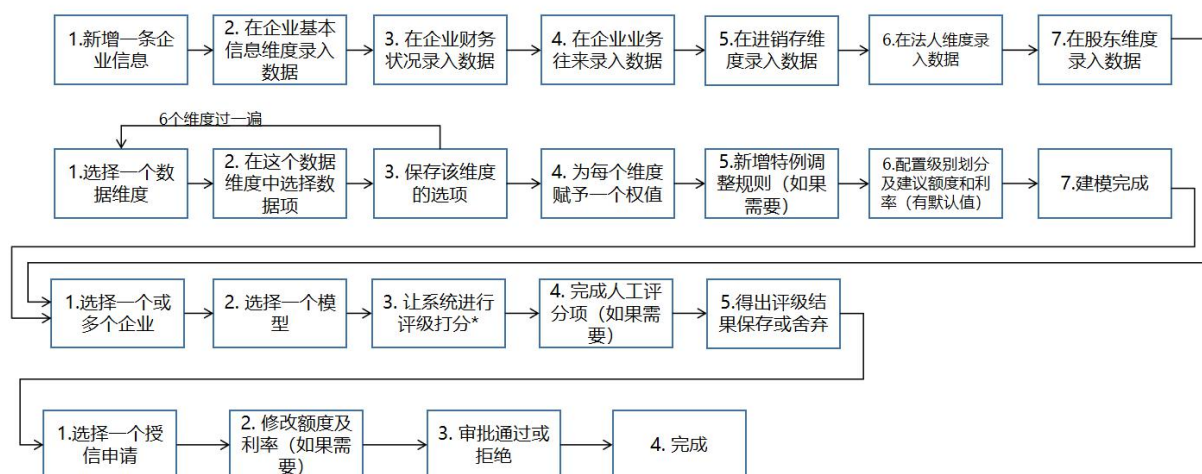


图 5-1 风控评级模型处理流程总括图

如图 5-1 所示，为本系统贷前授信风控评级模型处理流程图。一共四大模块，正如图上所示，分别为第一行的数据录入模块、第二行的建模录入模块、第三行的风险评级模块以及第四行的授信审批模块，每一行的数据流对应一个大模块的完整功能流程。

以下是本系统的第一个页面，登录页面。用户身份验证，本系统使用 Shiro 框架实现用户登录安全验证，SecurityManager 是 Shiro 框架的核心，属于典型的 Facade 模式，Shiro 通过该核心来管理内部组件实例，并通过它来提供各种安全管理的服务。

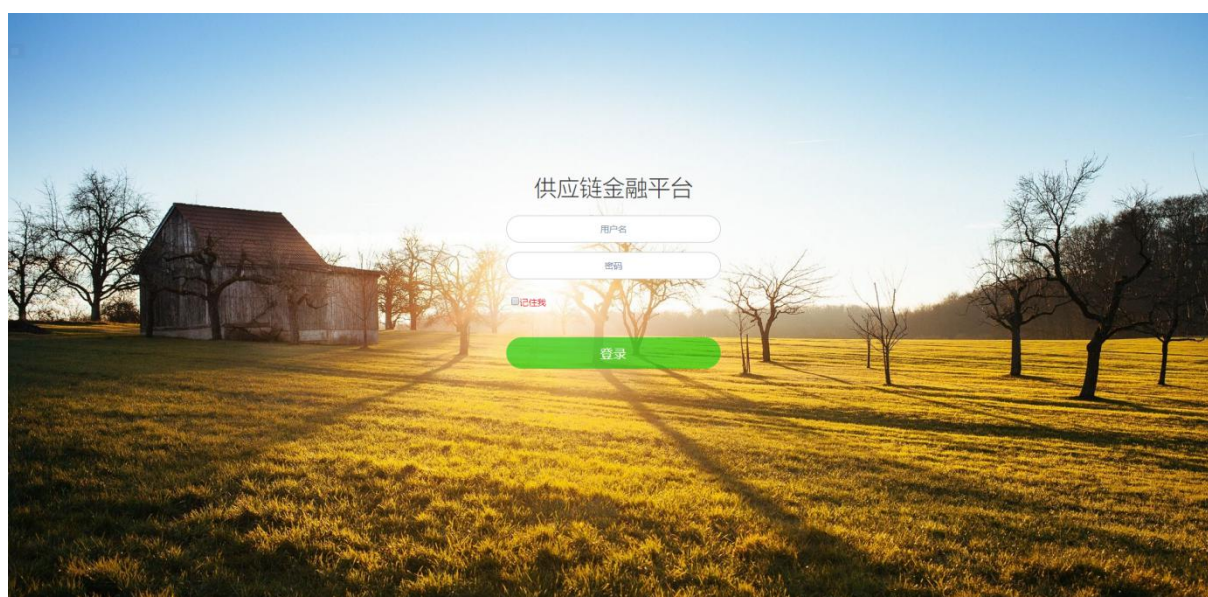


图 5-2 用户登录界面

以下为项目 Shiro 框架的主要配置代码：


```
<bean id="shiroFilter"
    class="org.apache.shiro.spring.web.ShiroFilterFactoryBean">
    <!-- Shiro 的核心安全接口,这个属性是必须的 -->
    <property name="securityManager" ref="securityManager" />
    <property name="loginUrl" value="/login" />
    <!-- 用户访问未对其授权的资源时,所显示的连接 -->
    <property name="filters">
        <map>
            <entry key="logout" value-ref="logoutFilter"/>
            <entry key="authc" value-ref="formAuthenticationFilter"/>
        </map>
    </property>
</bean>
<bean id="logoutFilter" class="org.apache.shiro.web.filter.authc.LogoutFilter">
    <property name="redirectUrl" value="/login" />
</bean>
<bean id="lifecycleBeanPostProcessor"
    class="org.apache.shiro.spring.LifecycleBeanPostProcessor"/>
<!-- 基于 Form 表单的身份验证过滤器 -->
<bean id="formAuthenticationFilter"
    class="org.apache.shiro.web.filter.authc.FormAuthenticationFilter">
    <property name="usernameParam" value="username"/>
    <property name="passwordParam" value="password"/>
</bean>
```

applicationContext-shiro.xml 部分代码

5.1 数据录入模块的实现

数据录入模块不仅是本系统最为基础的模块,同时也是不可或缺的一个模块,后续的各个模块都依赖于数据录入模块的结构设计和该模块录入的数据。建模录入模块需要根据此模块的结构字段设计来设计自身模块的结构;企业评级模块需要以完成数据录入的需融资企业完整信息来进行风险评级;授信审批则是对该以录入的企业数据进行最后的额度和授信费率确认。

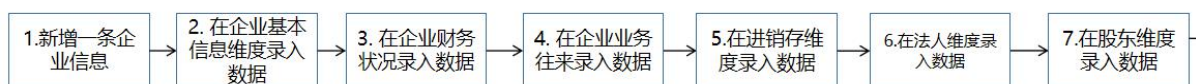


图 5-3 数据录入模块处理流程图

如图 5-2 所示,为数据录入模块处理流程图。从图中可知新增一条企业信息,总共需要在六个维度录入企业相应数据,六个维度分别为:企业基本信息、企业财务状况、企业业务往来、企业进销存数据、法人基本情况以及主要股东情况。

整个编码实现过程逻辑并不复杂,只是会有一些功能比较复杂难于实现的细节需要

用到的技术比较生疏，所实现的具体方式可能不是最简单的。下面将结合具体页面梳理数据录入模块的详细设计和实现细节。

(1) 数据录入列表页实现展示：



图 5-4 数据录入列表页

如图 5-4 所示，为数据录入页在前端框架 AngularJS 下实现后的实际样貌，看起来与第 4 章的原型设计图有许多出入，而我们必须知道的是使用 Axure RP 进行原型设计的初衷，只是为了提供给后期进行开发工作的程序员一个页面样式风格参考以及页面必需的功能项。

这里页面不仅比原型设计时美观简洁，且在每个字段下方加放了一个实时搜索框，减少了与后台的交互次数，从而提升了查询效率。界面的美观简洁依赖于 Bootstrap 样式，而实时搜索框就要归功于所选用的 AngularJS 框架。对于实时搜索框，只需在输入框中输入想要查询的内容，无论完整与否，都能快速检索过滤出来并显示在页面上，其实现原理是：在发起了访问数据录入列表页后，立刻向后台请求所有符合现实条件的企业数据，显示在页面并缓存在浏览器，于是在输入框中输入想要查找的内容回实时触发检索过滤功能，即时的过滤出符合查询条件的记录显示在页面，整个过程都由 AngularJS 缓存在浏览器中的相应 JavaScript 文件执行完成。比起以往一般的 Web 网页应用后台实现复杂的分页技术，该功能的实现极大的减少了后台开发的工作量和代码量，同时也极大程度的降低了前端网页与后台的交互次数，提升了整个系统的性能。

而且在 AngularJS 框架下，该小技巧极易实现，只需在 table 表单中加入一些指定的属性即可，下面是实现该技巧的相应示例代码：

```
<td data-title="'统一社会信用代码'" filter="{socialCreditCode: 'text'}"
sortable="'socialCreditCode'">{{enterpriseInformation.socialCreditCode}}</td>
data-list.html 部分代码
```

以上代码块是 data-list.html 的部分代码，展示的是图 5-4 中数据录入模块企业信息列表中“统一社会信用代码”数据列的 `<td>socialCreditCode</td>` 表单项。其中 `data-title="'统一社会信用代码'"` 的作用是给本数据列标题命名，比以往还要在表单前加一行表头的方式简单了许多；而 `filter="{socialCreditCode: 'text'}"` 则是下方实时搜索框实现的关键所在，指定了输入的内容将以 text 格式作为 socialCreditCode 字段在浏览器后台进行搜索过滤；且 `sortable="'socialCreditCode'"` 将根据 socialCreditCode 属

性的具体内容进行排序。另外，上述代码中 `<td>{{...}}</td>` 里面 `{{...}}` 格式的代码，是目前前端应用进行数据绑定最常见的形式，这就是“Mustache”语法（双大括号）的文本插值，`{{...}}` 标签将会被替代为对应数据对象上属性的值，在本例中该属性即是 `enterpriseInformation.socialCreditCode`。无论何时，绑定的数据对象上属性的值发生了改变，插值处的内容都会更新，这便是 AngularJS 强大的双向数据绑定特性。

（2）数据录入编辑页实现展示：

The figure displays a web interface for data entry and editing. At the top, there is a navigation bar with six tabs: 1. Enterprise Basic Information (selected), 2. Enterprise Financial Status, 3. Enterprise Business History, 4. Enterprise Inventory Data, 5. Legal Person Basic Information, and 6. Main Shareholder Situation. Below the tabs, the 'Enterprise Basic Information' form is shown. It contains several input fields: 'Unified Social Credit Code', 'Enterprise Name', 'Legal Representative' (with a sub-field for 'Name + ID Number'), 'Registered Capital (10,000 RMB)', 'Establishment Date', 'Operating Period (Years)', 'Office Location', 'Operating Scope' (with a 'View Details' button), 'Registered Capital Contribution Ratio (%)', 'Executive Director' (with a sub-field for 'Name + ID Number'), 'Supervisor' (with a sub-field for 'Name + ID Number'), 'Employee Count', 'Company Type', 'Trademark Registration Information', 'Intellectual Property出质登记信息' (with a 'View Details' button), and 'Branch Institution Count'. At the bottom of the form, there are two buttons: 'Return to List' and 'Next Step'.

图 5-5 数据录入编辑页

如图 5-5 所示，为数据录入编辑页，从数据录入列表页点击绿色新增按钮即可进入该页面。从图中可以看到，一共有六个选项卡，分别对应着数据录入模块输入企业信息的六大维度，在完成当前页内容的输入工作后，点击下一步即可进入到下一个选项卡进行其它的数据录入工作。

图中，我们可以看到有两个输入框是灰色带阴影的，框中内容是蓝色的“查看详细”，分别是经营范围和知识产权出质登记信息。再点击这类输入框后会弹出一个窗口，即可在弹出的窗口中录入相应信息，这里的弹出式录入的企业信息不会作为建模标准，即不会被选作系统的评级字段，而是作为长文本信息提供给审核员查看，审核员可以根据已经录入的此类信息对最终授信额度以及融资信息费率作出改变。以下是弹出框的实现方式，以点击知识产权出质登记信息为例，点击事件代码如下所示：

```
$scope.addIntellectual=function(){
    var dlg = dialogs.create(
        'intellectualInformation-edit.html','intellectualInformationController',
        {
            "intellectualId":$scope.intellectualInformation,
            "isAdd":isAdd,
            "intellectualInformation":intellectualInformation
        },
        {
```

```
        size: 'md',  
        'backdrop': 'static'  
    }  
);  
dlg.result.then(function(information){  
    intellectualInformation=information;  
    $scope.intellectualInformation=intellectualInformation.intellectualId;  
    toaster.pop('success', '', '数据录入成功, 请继续! ');  
});  
};
```

data-edit.js 部分代码

以上代码块是 data-edit.js 的部分代码, 当点击知识产权出质登记信息输入框时触发该函数, 使用 `dialogs.create(...)`; 创建一个弹窗页面, 定义对应的 Controller 页面。其后跟的参数中 `size: 'md'` 的作用是设置弹窗页面大小为中, 其可取值为 `sm`、`md`、`lg` 以及 `llg`, 表示意义如其字面意思, 分别是小、中、大以及超大; 而 `'backdrop': 'static'`, `backdrop` 值设置为 `static` 使窗口静态化, 效果是点击弹出窗体外的区域, 弹出窗体不消失, 不写这行代码则默认点击弹出窗体外的区域弹出窗体将消失。以下是弹出窗体的效果截图, 这里以点击知识产权出质登记信息时触发事件为例:

图 5-6 弹出窗体示例图

再往下看到的是回调函数: `dlg.result.then(function([param...]){...})`; 这里点击确定, 已经录入的信息作为参数回传给父级作用域 `$scope` 中的变量

`intellectualInformation`, 以进行下一步的数据入库操作。同时, `toaster.pop('success', '', '数据录入成功, 请继续!')`; 我们可以理解, 当数据录入成功时 (即关闭当前弹出窗口时并且数据录入过程顺利), 将在网页显示 `数据录入成功, 请继续!` 提示, 以下是 AngularJS 框架提供的 `toaster.pop` 基础样式示例图:



图 5-7 Success 示例图

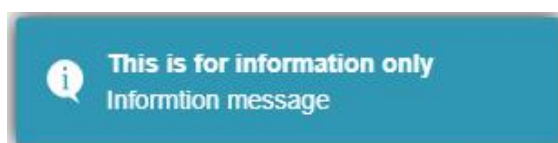


图 5-8 Information 示例图

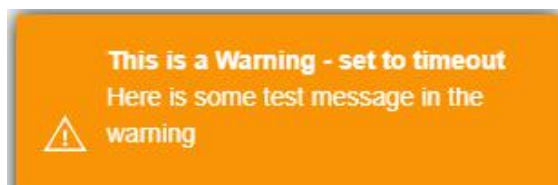


图 5-9 Warning 示例图

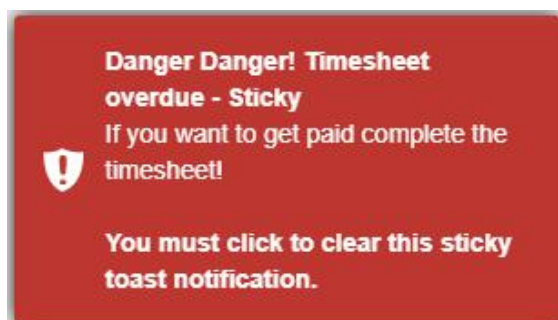


图 5-10 Danger 示例图

如图 5-7 到图 5-10 所示, 为 AngularJS 框架自带的提示框样式示例图, 分别代表四个级别的提示框样式。四个级别依次为 Success、Information、Warning 以及 Danger, 在本系统中未使用到 Information 级别的提示框, 在操作成功时使用 Success 提示框, 提示修改密码等情况时使用 Warning 提示框, 系统出现错误时 (例如: 404 NOT FOUND) 则使用 Danger 提示框。可在代码中自定义输入框的显示样式, 可设置显示时长 (自动消失时间)、点击即消失等操作, 既美观又提升了用户体验, 使得本系统的交互效果更加友好。

5.2 建模录入模块的实现

建模录入模块的设计依赖于数据录入模块的结构字段，但其在整个系统的结构上起的作用又与数据录入模块相辅相成，作为与数据录入模块平行的模块，与之共同形成本系统最基础的两大信息输入模块。

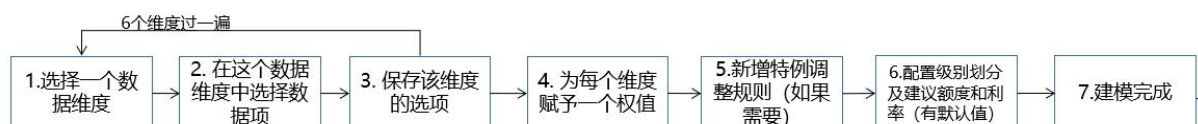


图 5-11 建模录入模块处理流程图

如图 5-11 所示，为建模录入模块处理流程图。从图中可以看出，本系统建立一个评级模型，除了要在数据录入模块的六大维度上设置标准值，还需要为最后的风险评级过程录入一些模型特有数据，分别是赋予不同维度权值、特例调整规则、配置级别划分以及设置不同级别的建议额度和利率。到此，一个完整的评级模型才算建成了，下面将展开叙述本模块的详细实现细节。

（1）建模录入列表页实现展示：

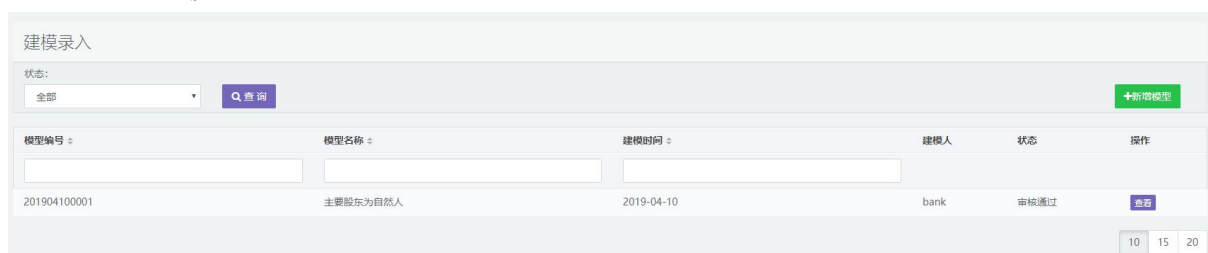


图 5-12 建模录入列表页

如图 5-12 所示，为建模录入模块的首页，也是该模块的列表页，整个页面风格和数据录入列表的页面相同。银行录入员登录本系统时，进入此页面后，默认查询状态为全部，提供新增、修改和查看功能入口按钮；银行审核员登录本系统时，进入此页面后，默认查询状态为等待审核，提供查看和审核的功能入口按钮。

（2）建模录入编辑功能评级划分页实现展示：

新增评级划分

1 评级划分

2 企业基本信息

3 企业财务状况

4 企业业务往来

5 企业进销存数据

6 法人基本信息

7 股东为自然人

8 股东为机构

9 赋予权值

10 特例调整规则

信用评级	评分参考值	信用评级系数(%)	建议利率(%)	建议信息费率(%)	建议最大期限(天)
A	80 <= 信 <= 100	90 <= 信 <= 100	<input type="text"/>	<input type="text"/>	<input type="text"/>
B	65 <= 信 <= 80	80 <= 信 <= 90	<input type="text"/>	<input type="text"/>	<input type="text"/>
C	55 <= 信 <= 65	60 <= 信 <= 80	<input type="text"/>	<input type="text"/>	<input type="text"/>
D	40 <= 信 <= 55	50 <= 信 <= 60	<input type="text"/>	<input type="text"/>	<input type="text"/>
E	30 <= 信 <= 40	30 <= 信 <= 50	<input type="text"/>	<input type="text"/>	<input type="text"/>
F	0 <= 信 <= 30	0 <= 信 <= 30	<input type="text"/>	<input type="text"/>	<input type="text"/>

基础信用额度*:

币种*:

模型名称*:

返回列表

下一步

图 5-13 建模录入评级划分页

如图 5-13 所示，为建模录入编辑功能的评级划分页。建模录入编辑功能在设计之初将评级划分页放在最后一个选项卡，但在后期代码实现过程中发现了这种做法的问题所在。因为评级划分页中需要输入此模型的基础信息，以区别其它模型，基础信息包括基础信用额度、币种以及模型名称，其中模型名称作为主要识别信息，于是此页面应排在第一选项卡，且将三个基础信息标上红星作为非空字段。

在这里，图中我们可以看到一个下拉框，选择币种信息，其可取值为全部、人民币以及美元。我们所熟知的经典实现下拉列表选择框的方式如下：

```
<select class="form-control" >
  <option value="">全部</option>
  <option value="01">人民币</option>
  <option value="02">美元</option>
</select>
```

经典下拉框示例代码

本系统实现过程中所使用的下拉框的各个选项值都作为数据字典值存放于数据库中，每次加载的页面包含下拉框时则向数据库请求相应值初始化至页面元素中，下面两个表格描述数据库中数据字典值得存放示例：

表 5-1 数据字典组表（部分内容）

id	ele_group_code	ele_group_name
33	(null)	币种

表 5-2 数据字典表（部分内容）

id	ele_group_id	ele_code	ele_name
----	--------------	----------	----------

93	33	01	人民币
94	33	02	美元

如上表 5-1 和 5-2 所示，分别为数据字典组表和数据字典表，在数据字典组表定义数据元素，在数据字典表中定义数据元素包含的内容值，这里给出的是币种元素的数据字典存放示例。

```
LeasingCommon.getDictByGroupId(33).then(function (result) {
    $scope.currencyTypes = result;
}, function (err) {
});
```

pingji-list.js 部分代码

以上代码块为 pingji-list.js 部分代码，在初始化评级划分页面时，该段代码会自动执行，访问数据库，`LeasingCommon.getDictByGroupId(33)`根据字典组 id 值找到相应字典存放内容，然后将结果作为参数回调给全局作用域变量 `$scope.currencyTypes`。

```
<select name="currency" ng-model="baseInformation.currency" required
    ng-options="option.id as option.name for option in currencyTypes">
    <option value='' >全部</option>
</select>
```

pingji-list.html 部分代码

以上代码块为 pingji-list.html 部分代码，`<option value='' >全部</option>`表示给下拉框赋初使显示页面显示值为“全部”，但是提交页面时的值为空；`required`表示在提交页面时该下拉框值不能为空或 NULL；而 `ng-options="option.id as option.name for option in currencyTypes"`是 AngularJS 框架提供的下拉框赋值方式，在上一代码块 pingji-list.js 部分代码中，页面初始化时给全局作用域变量 `$scope.currencyTypes` 赋值了，在页面会自动生成经典下拉框代码，并且该变量就以键值对的形式赋值给自动生成的下拉框元素。

(3) 建模录入编辑功能赋予权值页实现展示：

图 5-14 建模录入赋予权值页

如图 5-14 所示，为建模录入编辑功能的权值配置页。在页面相应的 JavaScript 代码中，会为每一个输入框进行校验，校验其值必须为小于 100 的正整数。并且 6 个维度的权值总和必须等于 100，否则在提交此页面的时候浏览器右上角会弹出如图 5-15 所示的 Warning 提示框。



图 5-15 权值页面 Warning 提示框

(4) 建模录入编辑功能特里调整规则页实现展示：



图 5-16 建模录入特例调整规则页

如图 5-16 所示，为建模录入编辑功能基于关键因素的特例调整规则页。在此页勾选上相应的规则，每个企业在使用此模型进行风险评级时，若该企业信息符合以勾选规则，则评分不会高过该规则项所定标准。目前本系统只设置了图中的 4 项特例调整规则，以后考虑为用户提供自定义规则项，以完善系统，使得系统的交互性和用户体验更加友好。

5.3 企业评级模块的实现

企业评级模块的实现依赖于前两大模块提供的已审核数据。评级的过程就是银行评级人员选择一家已经审核通过的企业的的数据，然后放到一个已经审核通过的评级模型中去计算得出结果。

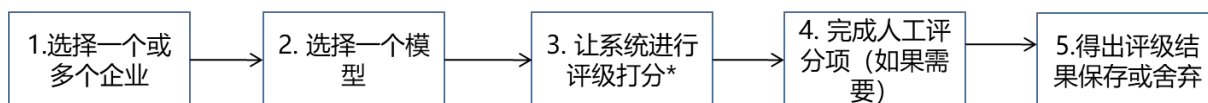


图 5-17 风险评级模块处理流程图

如图 5-17 所示，为企业评级模块数据处理流程图。本模块实现的关键核心部分在

于上图中的第三步：让系统进行评级打分。这里强调一下本系统的评级规则：

- ① 根据数据录入模块录入的企业数据起算每一个维度的总分；
- ② 根据每个维度的权重，计算出加权平均值；
- ③ 根据加权平均值对应出评级；
- ④ 根据该模型中生效的特例调整规则调整最后的结果。

其中第 ② 点加权平均值的计算可参考在本文“4.1.2 原型设计”中的详细示例描述，最后系统根据计算出的加权平均值匹配出模型中相应的评分等级，本系统中的评分等级一共包括 6 个等级，分别为 A、B、C、D、E、F，建模录入人员可自定义评分等级对应的加权平均值和信用评级系数。而信用评级系数是系统最后计算建议额度的关键数据，例如本系统给出的 B 级默认信用评级系数为(80%, 90%]，即小于等于 90%，大于 80%，若最后该企业在该模型下的评级结果为 B 级，假设该模型设置的基础信用额度为 2000000，则系统将给出的建议额度为：

$$[\text{基础信用额度} * 80\%, \text{基础信用额度} * 90\%] = [1600000, 1800000]$$

其计算方式的主要实现代码如下所示（以企业最后评分等级为 B 级为例）：

```
// 初始化 EnterpriseRating 对象
EnterpriseRating response=new EnterpriseRating();
// 获取模型的基础信用额度
BigDecimal blc=baseInformation.getBaselineCredit();
// 计算最小建议额度
BigDecimal pbMin=baseInformation.getbMinCreditRatingCoefficient().multiply(blc)
    .divide(HUNDRED,3,BigDecimal.ROUND_HALF_UP).setScale(2,
        BigDecimal.ROUND_HALF_UP);
// 计算最大建议额度
BigDecimal pbMax=baseInformation.getaMinCreditRatingCoefficient().multiply(blc)
    .divide(HUNDRED,3,BigDecimal.ROUND_HALF_UP).setScale(2,
        BigDecimal.ROUND_HALF_UP);
// 将最小、最大额度赋值给 response
response.setProposedAmountMin(pbMin);
response.setProposedAmountMax(pbMax);
return response;
```

FinalShowManager.java 部分代码

(1) 企业评级列表页实现展示：

企业评级

统一社会信用代码	企业名称	法定代表人	注册资本(万)	操作
91110105078556180Q	北京仁聚汇通信息科技有限公司	董洪亮	5555.56	评级

10 15 20

图 5-18 企业评级列表页

如图 5-18 所示，为企业评级列表页。由于每条企业数据可以选择不同的评级模型进行风险评级，故此列表页不存在状态相关字段，从而取消了该列表页的根据状态进行查询的输入框以及查询按钮，只保留了实时搜索框，实时搜索框具体讲解见本章第 1 节数据录入列表页实现展示。

(2) 企业评级评级页实现展示：

图 5-19 企业评级评级页

如图 5-19 所示，为企业评级模块的评级页，该界面形式为弹出窗口，评级人员根据要求选择用于评级的模型，若该模型已评过该企业，则网页右上角会弹出相应的 warning 提示框并将确定按钮变灰失效；若选用的模型未评过该企业，则可继续点击确定按钮完成评级操作。

5.4 授信审批模块的实现

在评级结果提交以后，授信审批人员即可在授信审批列表页，对于每一条授信申请，授信审批人员都可进行查看与审核操作。

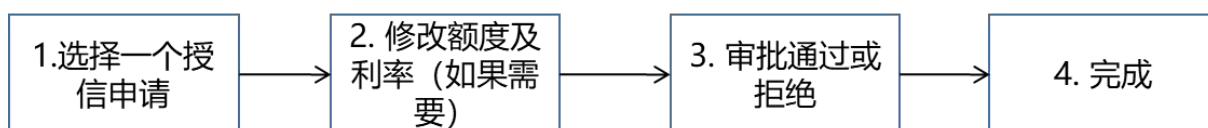


图 5-20 授信审批模块处理流程图

如图 5-20 所示，为授信审批模块数据处理流程图。授信审批人员的工作就是对每一条授信申请进行审批（通过或拒绝），其主要流程为：在列表页选择一条授信申请记录，点击审核按钮，而后进入授信审批详情页，在详情页修改额度以及利率（如果需要），最后输入审批意见（审核拒绝时必须输入），这样就完成了一次授信审批。

(1) 授信审批列表页实现展示：

授信申请审核

企业名称:

评级模型:

评级结果:

状态:

全部

全部

全部

全部

查询

统一社会信用代码	企业名称	评级模型	最后得分	评级结果	币种	授信额度	授信利率(%)	信息费率(%)	授信期限(天)	状态	操作
91110105078556180Q	北京仁聚汇通信息科技有限责任公司	1	0.00	F	人民币					等待审核	查看 审核
91110105078556180Q	北京仁聚汇通信息科技有限责任公司	主要股东为自然人	65.67	B	人民币	1,800,000.00	2.0000	3.0000	25	审核通过	查看 打印

101520

图 5-21 授信审批列表页

如图 5-21 所示，为授信审批模块的列表页。该页提供多种查询方式，包括按企业名称查询、按评级模型查询、按评级结果查询以及按审批状态进行查询，也可同时输入多个条件进行精确查询。列表页为每条未审批或审批拒绝授信申请记录提供查看和审核功能入口按钮，为每条审批通过授信申请记录提供查看和打印功能入口按钮。

(2) 授信审批详情页实现展示：

统一社会信用代码:	91110105078556180Q						
企业名称:	北京xxxx公司						
评级模型:	主要股东为自然人						
最后得分:	65.67分						
评级结果:	B						
建议额度:	[1,600,000.00 , 1,800,000.00]						
建议利率(%):	2.0000						
建议信息费率(%):	3.0000						
建议期限(天):	25						
币种:	人民币						
分项得分:	<table><tr><td>企业基本信息=70.00(10%)</td></tr><tr><td>企业财务状况=80.00(30%)</td></tr><tr><td>与企业业务往来=30.00(20%)</td></tr><tr><td>企业进销存情况=53.33(20%)</td></tr><tr><td>企业法人情况=90.00(10%)</td></tr><tr><td>主要股东情况=90.00(10%)</td></tr></table>	企业基本信息=70.00(10%)	企业财务状况=80.00(30%)	与企业业务往来=30.00(20%)	企业进销存情况=53.33(20%)	企业法人情况=90.00(10%)	主要股东情况=90.00(10%)
企业基本信息=70.00(10%)							
企业财务状况=80.00(30%)							
与企业业务往来=30.00(20%)							
企业进销存情况=53.33(20%)							
企业法人情况=90.00(10%)							
主要股东情况=90.00(10%)							
授信额度:	1,800,000.00						
授信利率(%):	2.0000						
授信信息费率(%):	3.0000						
授信期限(天):	25						
审核状态:	审核通过						
审核意见:	11						

图 5-22 授信审批详情页

如图 5-22 所示，为授信审批模块审批详情页。该页被本系统多处复用，分别是评级完成后展示评级结果页、授信申请查看页、授信申请审核页以及最后的授信申请打印界面。

另外，在本模块中由于需要展示金额信息，并且经常涉及到大额的货币数量展示。于是也应用到了 AngularJS 框架的过滤器，过滤器可以使用一个管道字符（|）添加到表达式和指令中，AngularJS 过滤器可用于转换数据，在本模块中主要运用过滤器将数字格式化为货币格式的数据展示在页面上。以本页中的授信额度行为例，其实现代码如下：

```
<tr>
  <td class="col-dt-k">授信额度:</td>
  <td class="col-dt-v">{{creditDetail.creditLine | currency: '' : '2'}}</td>
</tr>
```

creditReview-detail.html 部分代码

其中{{creditDetail.creditLine | currency: '' : '2'}}就使用了管道过滤器，其功能是将数字格式化为货币格式的数据。其一般格式为：{{ currency_expression | currency : symbol : fractionSize}}，symbol（可选）表示要显示的货币符号或标识符，例如'¥'或'\$'，fractionSize（可选）整数的小数点的数目，默认为当前区域的默认最大大小，本模块中设置该值大小为 2，即最后页面显示数据时将显示两位小数。

结论

（1）工作总结

经过这一段时间曲折的设计与开发，本供应链金融平台贷前授信评级子系统的基础功能要求已经基本实现，可作为银行或金融机构的对需融资的中小企业进行线上风险评级并给出建议额度与利率的系统平台。本系统后台使用 Java 语言进行开发，使用的框架是市面公司常用的 SSM（Spring+Spring MVC+MyBatis）框架，按照分层开发的架构，使得整个系统后端架构井然有序，极大提高了开发编码效率；前端框架是功能强大的 AngularJS，在实现页面展示上、搜索优化上、数据格式化显示上以及实现一些页面逻辑上发挥了其强大作用；在进行系统开发之前使用 Axure RP 进行了原型设计，快速、高效的创建原型，为系统后期开发指明了方向。相比传统 Web 应用，本系统放弃了传统的 JSP 视图层的显示的开发方式，使用前端框架 AngularJS 对前端页面进行组件化方式的开发，同时，经过对此次毕业设计的开发工作，我也体会到了更好的 Web 思想：Web Components，未来迟早要到来，Web Components 是趋势。

（2）对当前本 Web 应用的个人看法

首先，本系统基础功能虽然基本已经实现，但是许多功能需要支持定制服务本系统目前无法提供，例如建模录入模块中的特例调整规则页，以后需要考虑为用户提供自定义规则项。其次，从软件生命周期的方向来看，本 Web 应用还需不断的进行市场调研，使数据录入模块甚至所有模块不断更新，才能走向线上，成为一个标准的互联网平台化产品。

致 谢

本次毕业设计的顺利完成，首先感谢东华理工大学软件学院对我四年来的培养，使我有能力独立完成本系统的设计与开发。同时，在系统实现过程中，感谢同学们热心的技术援助和方案贡献，使我充满信心的完成开发。

特别的，要感谢我的指导老师，张军老师。正是他严肃的科学态度、严谨的治学精神以及在时间上的严格把控，才使得本毕业设计顺利完成，同时，张军老师阶段性对我的工作给出的建议，也使得我少走了很多弯路。在此，感谢张军老师的悉心指导。

参考文献

- [1] 闫俊宏, 许祥秦. 基于供应链金融的中小企业融资模式分析[J]. 上海金融, 2007(02):14-16.
- [2] 冯静生. 供应链金融:优势、风险及建议[J]. 区域金融研究, 2009, 7(2):51-52.
- [3] 深圳发展银行、中欧国际工商学院“供应链金融”课题组:《供应链金融:新经济下的新金融》, 上海, 上海远东出版社, 2009.
- [4] Timme S, Williamstimme C. The financial-SCM connection[J]. 2000.
- [5] Hofmann E. Supply Chain Finance: some conceptual insights[J]. 2005.
- [6] Brad Green,Shyam Seshadri.用 AngularJS 开发下一代 Web 应用[M].电子工业出版社, 2013.
- [7] 赵野望, 徐飞, 何鹏飞. AngularJS 权威教程.人民邮电出版社, 2014.
- [8] 易剑波. 基于 MVVM 模式的 WEB 前端框架的研究[J]. 信息与电脑(理论版), 2016(19):76-77+84.
- [9] 培学网 ANDY 编. AXURE RP8.0 入门宝典 网站和 APP 原型设计实战. 人民邮电出版社, 2016.07.
- [10] 苟文博, 于强. 基于 MySQL 的数据管理系统设计与实现[J]. 电子设计工程, 2017, (06): 62-65.
- [11] 王晓华. 试析 MySQL 数据库性能的调优[J]. 电脑编程技巧与维护, 2016, (22): 48+82.
- [12] 田珂, 谢世波, 方马. J2EE 数据持久层的解决方案[J]. 计算机工程, 2003, 29(22): 93-95.
- [13] 张宇, 王映辉, 张翔南. 基于 Spring 的 MVC 框架设计与实现[J]. 计算机工程. 2010(04).
- [14] 徐雯, 高建华. 基于 Spring MVC 及 MyBatis 的 Web 应用框架研究[J]. 微型电脑应用. 2012(07).
- [15] 杨开振. 深入浅出 MyBatis 技术原理与实战. 电子工业出版社, 2016.09.