### 2.2需求分析

### 2.2.1功能需求分析

### 2.2.2非功能需求分析

### **2.3数据流程分析**

数据有前端页面通过用户进行注册或者填写信息，通过接口发送到后端，再由后端经过数据处理保存到数据库种。

# 3总体设计

### **3.1总体功能设计**

### **3.2开发环境与工具**

## 3.2.1 硬件环境

本项目的系统硬件开发工具是个人笔记本电脑，其配置是CPU:英特尔酷睿i7四核处理器，运存：16G，硬盘：1T。同时本项目的开发规模较小，可在一般配置下的电脑均可运行。

## 3.2.2 软件环境

对于后台SpringBoot项目可在 windows7以上的操作系统下运行。整体的编程使用java语言，因此需要java的编译和运行环境。本项目最终部署到了阿里云的云服务器中，在服务器中同样也需要Java的编译和运行环境。

## 3.2.3 系统开发工具

本次开发代码的编写主要是在Windows系统的 实现，后端代码编写使用了IntelliJ IDEA 2020.3.2开发工具，Web网站前端的代码编写在WebStorm软件上实现，接口测试主要使用了postman，此外，Mysql数据库的可视化软件使用SQLyog。同时使用了MobaXterm来访问阿里云服务器终端。

# **4详细设计**

### **4.1 主要功能模块设计**

### **4.1.1 主界面设计**

## 2.1 Web前端架构设计

前端设计为侧边栏布局形式，采用 Html5 和 JavaScript 实现。Js 效果主要采用了 JQuery 框架和art-template模版引擎数据响应，整体布局采用了 BootStrap响应式框架框架。前端的关键实现部分有以下几点:

(1)折线图,环状图,柱形图的实现通过jQuery的$.ajax获取后端接口中的数据。

step1: 引入jQuery

<script type="text/javascript " src="vendors/jquery-1.11.2.min.js "></script>

step2:发送请求获取后端接口数据

$.ajax({

method: 'GET',

url: 'http://119.23.241.76:8000/employer/department/select/?department\_name=&company\_name='+localStorage.getItem('company\_name')+'&random='+Math.random(),

dataType:"json",

success: function (res) {

var htmlStr = template('tpl-table', res)

$('tbody').html(htmlStr)

}

})

（2） 主页中采用 echarts 插件展示数据显示图表。

引入 echats.js 后，获取页面元素，进行对象的初始化。再设置图表的样式和数据，最后渲染到 html 页面。

（3） 主页中采用 art-template模版引擎拼接数据和页面样式。

step1:引入art-template

<script type="text/javascript" src="./vendors/template-web.js"></script>

step2:编辑模版和页面拼接方式

step3:获取数据并与模版拼接

（4） 数据分类功能

前端向Spring boot后台发送请求，后台根据前端的参数在数据库中获取数据再返回给前端进行渲染。在请求时，主要采用 Jquery 封装的 ajax 方法实现。

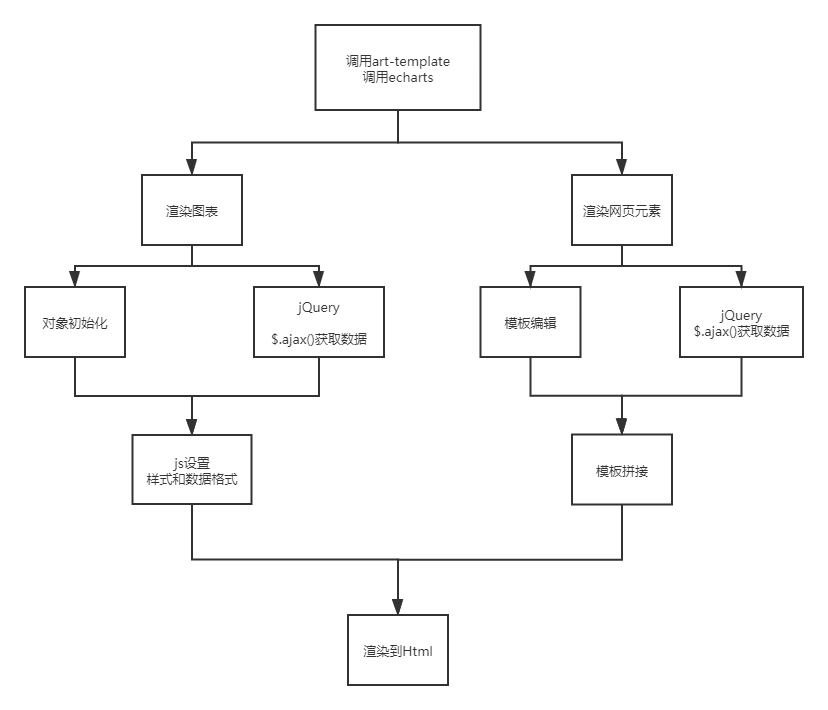


图 1网页渲染流程图

## **2.2** Web后端技术设计

Web 后端使用 SpringBoot 架构搭建，从Mysql 中获取数据。使用 MyBatis对 Mysql 数据库操作，存取数据。MyBatista是一个半自动化的ORM框架（对象关系映射）。

MyBatista优点：

* 无侵入：只做增强不做改变，引入它不会对工程产生影响。
* 损耗小：启动即会自动注入基本的CURD，性能基本无消耗，直接面向对象操作。
* 强大的CRUD：内置通用Mapper、通用Service。
* 支持Lambda形式调用：通过Lambda表达式，方便的编写各种查询条件。
* 支持主键自动生成。
* 解除sql和程序代码的耦合：通过提供DAO层，将业务逻辑和数据访问逻辑分离，便于维护提供xml标签，支持编写动态sql。

后端技术架构图如下图 2所示。

MyBatis

DAO层

MySQL

Service层

Controller层

Springboot

图 2 web后端技术架构图

结合系统功能，设计出 SpringBoot 后台框架用于实现数据库与前端页面的数据交互，为系统提供了稳定、易用的数据交互接口。SpringBoot 是 Java 平台 上的一种开源应用框架，用来简化新 Spring 应用的初始搭建以及开发过程。约定优于配是一种由 SpringBoot 本身来配置目标结构。系统采用的 SpringBoot 架 构设计如图3 所示。

PageMapperper

ReqMapperper

BasicMapperper

数据层

Mapper

Springboot技术栈

PageServicevper

ReqServiceper

BasicServiceper

数据逻辑实现层 Service

MyBatista

PageController

ReqController

BasicController

控制层Controller

Spring MVC

Web 前端

图3 SpringBoot 架构设计图

（1）数据层 Mapper

数据层中包含了对数据的所有操作，使用简单的注解用于配置和原始映射，将接口和Java的POTJOS（Plain Old Java Objects，普通的Java对象）映射成数据库中的记录。前端所需求的数据都是通过Mapper层来获取到原始数据然后返回到Service层的。该层中包含有几十条的SQL 语句，用于对数据库的CRUD，已到达目的所求。

（2）数据逻辑实现层 Service

数据逻辑实现层是从Mapper层中获取到原始的数据，再编写实现功能的逻辑代码，以满足前端页面所需求的数据类型，数据格式。该层包含着所有的逻辑实现功能的代码（是后端技术的核心层次），方便了Controller层的调用和简洁！

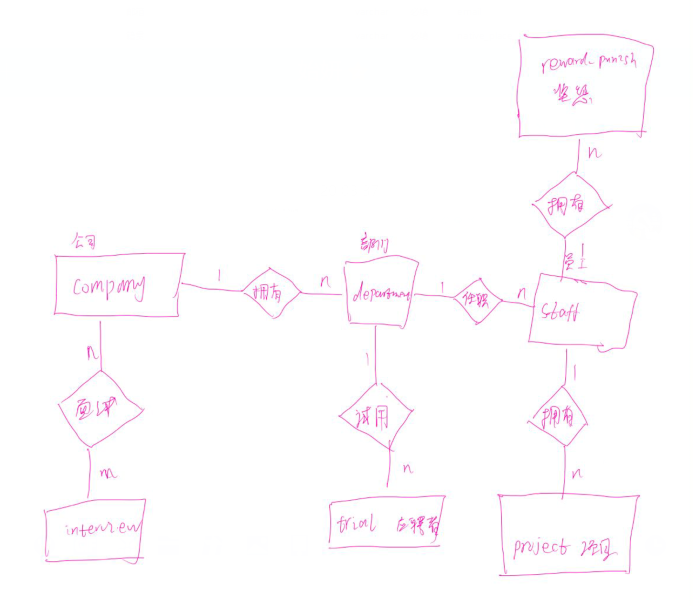
（3）控制层 Controller

控制层直接与页面进行数据交互，前端可以从该层中获取到所有需要的数据，以便将数据展示在页面上。该层时将所有的接口展示出来，并且无任何的逻辑处理，已达到高效和简洁的目的。

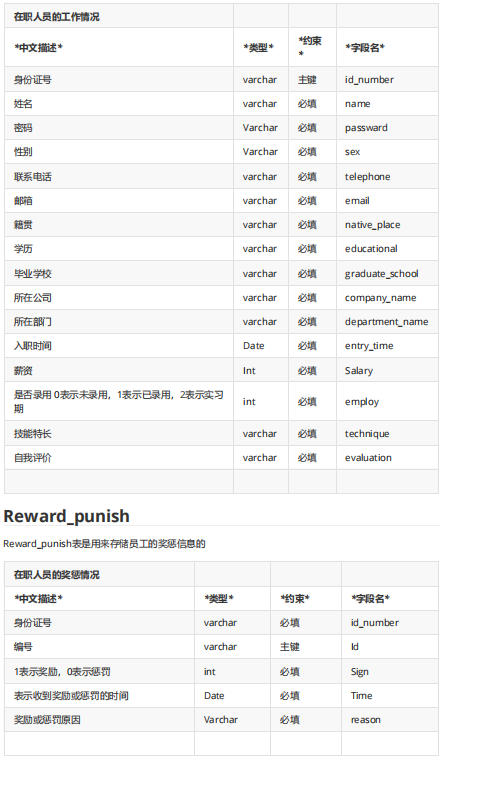
### **4.2数据库设计**

### **4.2.1 概念结构设计**

### **4.2.2 逻辑结构设计**











### **4.2.3 物理结构设计**

# **5系统实现**

### **5.1用户登录模块**

### **5.2 主界面模块**

### **5.3 检索与统计功能**

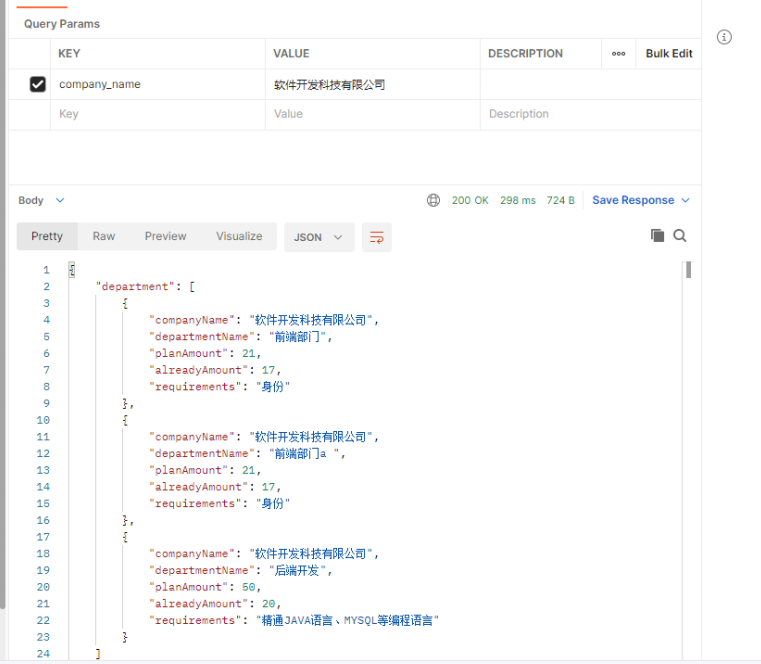
### **5.3.1检索功能实现**

### **5.3.2统计功能的实现**

# **6 系统测试与运行**

## 6.1后端接口测试

http://119.23.241.76:8000/employer/department/select/?company\_name=软件开发科技有限公司



## **6.2 前端页面测试**