

layout	title	subtitle	date	author	header- img	catalog	tags	
post	Sig文档 整理平 台搭建 记录	so much problem!	2020-01- 05 23:00:00	Doni Daniel	img/post- bg- rwd.jpg	true	技 术	服 务 器

# Sig文档整理平台搭建记录

## 前言

本记录拟用生命周期法的方式来记录“Sig文档整理平台”（以下简称该平台）整个开发过程，仅记录了2月7日及以后关于该平台的相关事项

TencentCloud的使用记录不适合记录开发中的小点和规划。所以就由该文档来记录具体各container的内部细节，也是我的毕业设计过程的一个记录。

## 系统规划阶段

### 系统开发背景

在当今社会，拥有几个G的数据非常常见，集中式的数据爆炸后，开始向个人用户辐射。个人文档数量明显增加，然而个人用户有效文档占总文档的比重在不断减小，其中重要的原因就是个人用户对于文档的有效整理与重用率不高。能够快速查找到希望找到的文档，能够根据个人个性化的整理文档，成为一个亟待解决的问题。

在个人使用系统的过程中，文档的合理分类将占据文档使用的时间的一半，多次的修改文档的整理方式。使用时，会忘记文档位置与关键词，全局关键词搜索受限。在团队合作时，沟通交流是必不可少的，说明自己所掌握的知识库并了解团队成员的知识库，是一个团队不断前进的方法。因此，一款能将个人文档系统整理，个性化整理，团队文档与知识库整理的系统是不二选择。

### 系统的目标

1. 方便广大办公人士对工作生活文件进行分类，并帮助其提取知识与相关知识文件的寻找
2. 方便团队或组织对团队文档进行管理，帮助团队内部成员进行团队共同知识的学习

## 初步调查: 明确系统开发的目标和规模

### 明确目标:

1. 实现个人用户文档的有序管理,辅助用户进行文档的整理分类工作
2. 辅助用户对文档进行知识点提取与知识分享
3. 实现团队文档的合理管理, 辅助团队知识体系管理

### 规模:

可访问到互联网的对文档整理有需求的人群

受个人知识限制, 当前无法评估系统规模, 但预估应能承受500-1000人的并发使用

### 普通用户

1. 对个人文档进行上传、下载、分享与删除
2. 系统辅助进行分类
3. 根据文档知识需求能检索相关文档
4. 组建、加入和退出团队

### 团队管理者

1. 团队成员功能及以下
2. 当前团队内成员进行管理
3. 能发信息或提醒团队内成员

### 团队成员

1. 对当前团队成员公开的文档下载
2. 能获取团队的实时动态
3. 能发表自己的动态

## 可行性研究

### 分析系统开发的必要性

1. 为解决个人毕业需求所做毕业论文的材料
2. 为学习相关开发步骤和开发阶段，以用到实际公司的开发过程中
3. 解决在实际文档工作中，文档整理问题

## 技术可行性

1. 本人已对该系统的技术要点进行了实地的实验与学习，已能初步的进行系统的实际编码，并能实现部分需求
2. 能确实的解决一部分人的文档的整理工作

该网站的开发是建立在成熟可靠的技术基础上的，拟以MySQL为数据库，redis数据库加速。应用html、css、javascript等语言，结合vue、Django框架完成该网站的前端、后端以及数据库部分的开发，使用Nginx和Apache实现动静分离从而保证建立一个有效运行、功能完善的在线平台。

## 操作可行性

该网站的页面设计欲以简洁、直观为主，功能设计欲以实用、便捷为主，操作简单，功能完整，对于任何一个文件分类和知识共享需求的个人或群体来说，只要掌握简单的网站操作便可以轻松驾驭该网站所提供的所有功能模块，花费少量的时间便能够对网站的各种操作得心应手。

## 社会可行性

该网站将文档分类和知识共享相结合，能够促进各类群体内部的知识通过文档的形式在各个成员以及外界环境中进行传播，从而提高团队协作的效率、丰富组织的知识库、提高企业的核心竞争力。

## 初步方案

本部分文档撰写于2020年2月7日

	2020年			二月			
周日	周一	周二	周三	周四	周五	周六	
2	3	4	5	6	7	8	本周完成整体系统的规划
9	10	11	12	13	14	15	本周完成整体系统的分析与设计
16	17	18	19	20	21	22	前端后台数据库数据连接的实现
23	24	25	26	27	28	29	前端三级框架的实现
30							
	2020年			三月			
周日	周一	周二	周三	周四	周五	周六	
	1	2	3	4	5	6	团队增删改查功能实现
7	8	9	10	11	12	13	
14	15	16	17	18	19	20	
21	22	23	24	25	26	27	
28	29	30	31				
	2020年			四月			
周日	周一	周二	周三	周四	周五	周六	
				1	2	3	
4	5	6	7	8	9	10	
11	12	13	14	15	16	17	
18	19	20	21	22	23	24	
25	26	27	28	29	30	31	

## 1.6日原计划

1. 三月中旬就要去实习了，所以想在三月中旬前写出来程序，算下来一共11周时间，去掉过年的一周，就剩下10周时间了，五个模块10周，两周一个模块。

2. 当前版本是V1.1.4标志着第一版程序的第一个功能的第4次进步。

3. 从1.6-1.19开发第一个功能模块（文档管理模块）

## 第一阶段

1. 初步搭建B/S模型：一方面减少因系统带来的开发问题，另一方面减少因软件平台而带来的使用限制性
2. 完成系统的前端后台的基本功能

### 实际情况

1. 框架搭建完成，登录模块，文档管理模块，团队管理模块大体完成
2. 知识库生成模块与互动模块未完成

## 第二阶段

回顾大体功能，并总结相关技术点，查阅相关文章文献，为撰写毕业设计说明书做准备

检测系统bug，上线进行测试，并修复相关bug

## 第三阶段

撰写毕业设计说明书

# 系统分析阶段

---

## 一、明确用户的信息系统需求

### 1. 用户注册与登录功能

- ☒ 1) 使用邮箱进行用户标注与识别注册
- ☒ 2) 记录用户登录状态
- ☒ 3) 能使用邮箱和用户名进行登录

### 2. 文档管理与自动分类

- ☒ 1) 用户可上传文件，并做提交tag
- ☒ 2) 实现下载与删除功能
- ☐ 3) 文件压缩存储
- ☐ 3) 支持自动个性化分类

- ☐ 4) 支持少量的代码文档类型
- ☐ 5) 支持注册用户在网站上创建知识库或文档，并存储到服务端数据库
- ☐ 6) 公开文档会在网站内展示给所有用户
- ☐ 7) 提供站内搜索，迅速检索相应用户、标题或内容的文档
- ☐ 8) 可以对感兴趣的内容进行关注订阅或收藏
- ☐ 9) 用户可随时对文件进行编辑
- ☐ 10) 可以为文档和知识库增加标签（即关键字）
- ☐ 11) 可以向用户自动推送关注的用户最近更新的知识库
- ☐ 12) 支持识别用户上传的文档类别并归类存储
- ☐ 13) 支持多种文档的不同分类方式

### 3. 好友管理

- ☒ 可以搜索用户并可以关注网站上的其他的用户
- ☒ 可以给关注或粉丝留言

### 4. 团队协作功能

- ☒ 1) 注册用户可在网站创建、加入、退出团队
- ☐ 2) 用户可设置文档或知识库的权限，使文档或知识库仅选择的团队内部可见
- ☐ 3) 团队成员可集体对文档和知识库进行在线编辑、管理
- ☐ 4) 用户已加入的团队及其文档会在用户主页单独显示
- ☐ 5) 设置讨论区块，团队创建者可以发布issue，让团队成员能够集中讨论工作中遇到的问题，交流经验

## 二、系统详细调查与分析

### 金山文档概况分析

#### 组织结构图

企业拥有者（超级管理员）	
系统管理员（普通管理员）	部门管理员
普通用户	
文档	

管理功能图

金山系统											
企业主页											
团队管理		团队文档（类似个人）			成员管理		后台管理				
创建	...	分享与协作	标记	...	邀请	审批	概况	组织架构	订单管理	设置	

金山系统											
个人首页											
文档管理						消息管理			应用管理		
新建	最近	星标	文档目录	共享	搜索	提到你	文件助手	成员申请与提醒	日历待办	通讯录	表单

发现现行系统存在的问题

暂未发现明显问题

三、提出新系统逻辑方案

提出新系统的改进方案

业务流程图

数据流程图

[0层图](#) [1层图](#)

数据字典

表名	User
----	------

### 三、系统设计阶段

#### 总体结构设计

Public	Docker net [daniel-net]					
Vue-Nginx 前端管理服务器			Apache-Django 后台管理服务器		数据库服务器	
前端框架	静态页面处理	动态部署	页面生成	缓存	Redis	
Vue	Nginx	Apache	Django	磁盘	Mysql	
BASE	vue-nginx	apache-django		redis	mysql	
各版本及实现功能						
V1.0.1	支持vue	支持HCJS		支持本地服务		
V1.0.2	自动化部署			文件存储与用户存储		
	支持动静分离					
V1.0.3	支持HTTPS	链接并调用数据库				
V1.1.3	基本前端页面跳转	文档的接收与存储				
V1.1.4	文档上传、下载与删除	文档的接收与存储				
V1.1.5	手机端显示问题					
V1.1.4	多文档上传	多文档的接收与存储				

#### 详细设计

- 1. 组织机构与功能分析
- 2. 代码设计
- 3. 数据库逻辑与物理设计



4. I/O设计

5. 模块结构与功能设计

- 1. 用户注册登录模块：用户注册、登录和密码找回
- 2. 文档管理模块：文档的新建、编辑、发布、搜索以及删除
- 3. 知识库管理模块：知识库的建立、搜索、删除以及目录管理
- 4. 团队管理模块：团队的建立、搜索、删除以及讨论区主题的发布
- 5. 互动模块：热门文章和知识库浏览、关注、收藏以及参与话题讨论
- 6. 系统模块：个人信息编辑、用户管理、退出

6. 设计方案报告

四、系统实施阶段

编程记录

前端Vue框架

代码目录结构

docker-vue / src					
App.vue	assets	components	main.js	router	Vuex

router

index.js	/		
	/index	"	
		/signIn	"
	*		
home.js	/home	"	
		/friends	"
			talk
			find
fileManager.js	/fileManager	"	
		files	
team.js	/team	"	
		/worker	
		/maker	
		/manager	
knowNet.js	/knowNet	/	
setter.js	/setter	/	

components

public	Header.vue		
	Footer.vue		
	Error.vue		
index	Index.vue		
	views	index.vue	
	cards	signIn.vue	
		signIng.vue	
		signUp.vue	

home	Home.vue		
	views	home.vue	
	cards	userMenu.vue	
		fileList.vue	
		showFiles.vue	
		upLoadCard.vue	
	friends	friends.vue	
		views	friends.vue
		cards	friendsMenu.vue
			friendInfo.vue
			talkBox.vue
fileManager	FileManager.vue		
	views	fileManager.vue	
	cards	cloudFiles.vue	
		starFiles.vue	
		shareFiles.vue	
teams	teams.vue		
	views	teams.vue	
	cards	teamMaker.vue	
knowNet	knowNet.vue		
	views	knowNet.vue	
	cards		
setter	setter.vue		
	views	setterView.vue	
	cards		

## 原因

因为bootstrap依赖jquery，所以在使用npm安装bootstrap前先安装jquery 又因为jquery依赖popper.js，所以在安装jquery之前先安装popper 所以安装过程分为三步：1. 使用npm安装popper。2. 使用npm安装jquery。3.使用npm安装bootstrap

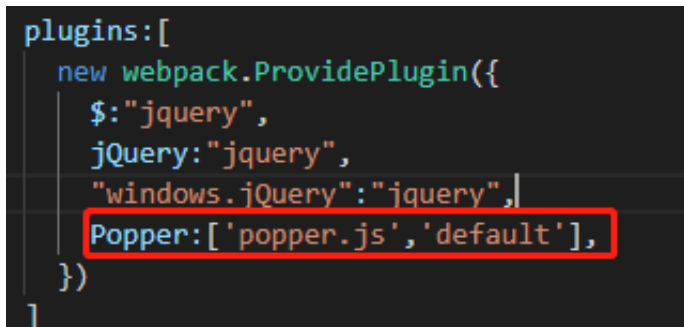
## 一、使用npm安装popper

1. 在build目录下的webpack.base.conf.js中加入

```
const webpack = require("webpack");
```

2. 在webpack.base.conf.js中module.exports找到plugins（如没有就创建），并添加上

```
plugins: [  
  new webpack.ProvidePlugin({  
    Popper: ['popper.js', 'default'],  
  })  
]
```



```
plugins:[  
  new webpack.ProvidePlugin({  
    $:"jquery",  
    jQuery:"jquery",  
    "windows.jQuery":"jquery",  
    Popper:['popper.js', 'default'],  
  })  
]
```

3. 使用npm安装popper

```
$ npm install popper.js --save-dev
```

## 二、使用npm安装jquery

1. 在package.json文件中找到"dependencies"项，并添加以下代码到其中：

```
"jquery": "^2.2.3"
```

2. 在build目录下的webpack.base.conf.js中，在module.exports中找到resolve，在其alias项中添加：

```
'jquery': 'jquery',
```

在module.exports中找到plugins，并添加以下代码：

```
plugins: [  
  new webpack.ProvidePlugin({  
    $: "jquery",  
    jQuery: "jquery",  
    "windows.jQuery": "jquery",  
    Popper: ['popper.js', 'default'],  
  })  
]
```

3. 在main.js中加入以下代码：

```
import $ from 'jquery'
```

4. 使用npm命令安装

```
$ npm install jquery@2 install jquery@2.2.3 --save-dev
```

### 三、安装bootstrap

1. 使用npm命令安装

```
$ npm install bootstrap@3 -S
```

2. 在需要使用的页面导入bootstrap.min.css/bootstrap.js就可以了。

```
import 'bootstrap/dist/css/bootstrap.css'  
import 'bootstrap/dist/js/bootstrap.js'
```

### 四、vue知识点

如果在想要在router-link上添加事件的话需要@click.native这样写

VueX 两个变量依次提交修改，监听模板的执行顺序与watch函数定义顺序有关

## 后台Django框架使用记录

# MySQL与redis使用记录

## TensorFlow使用记录

### 创建并配置tensorflow虚拟环境

---

#### 1. 创建虚拟环境

```
conda create -n TF2.0 python=3.6
```

#### 1. 初始化

```
conda init bash
```

#### 1. 进入虚拟环境

```
conda activate TF2.0
```

#### 1. 添加清华源为下载源

```
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/main/
conda config --set show_channel_urls yes
```

#### 1. 安装cuda10.0

```
conda install cudatoolkit=10.0
```

#### 1. 安装cudnn7.5.1

```
conda install cudnn=7.5.1 -c https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/main/
```

#### 1. 安装Tensorflow2.0

```
pip install tensorflow==2.0.0 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

1. 由于目前anaconda不支持tensorflow2.0的安装，只能用pip安装。测试环境搭建是否成功 运行 demo.py (python demo.py)，观察输出的tensorflow版本及是否使用GPU。

demo.py

```
import tensorflow as tf
version = tf.__version__
gpu_ok = tf.test.is_gpu_available()
print("tf version:",version,"\nuse GPU",gpu_ok)
```

docker pull tensorflow/tensorflow:2.0.0a0-gpu-py3

## gRPC

2. 调试
3. 测试
4. 试运行

## 五、系统运行阶段

---

1. 运行管理
2. 评价
3. 分析仍存在的问题