|  |  |
| --- | --- |
| 文档名称 | NLPPlat一期设计 |
| 文档版本 | 共 页 |
| 0.1 |

系统设计

|  |  |  |  |
| --- | --- | --- | --- |
| 拟制 |  | 日期 | 2020-09-12 |
| 评审人 |  | 日期 |  |
| 批准 |  | 日期 |  |

修订记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 日期 | 修订版本 | 修订章节 | 修订内容 | 作者 |
| 2020-09-12 | V0.1 | 全部 | 创建文档 | 程琨 |

目录

[1总体设计 4](#_Toc50899933)

[1.1 模块划分 4](#_Toc50899934)

[1.1.1 过程管理模块 4](#_Toc50899935)

[1.1.2 数据管理 5](#_Toc50899936)

[1.1.3 图形界面 5](#_Toc50899937)

[1.2 数据库设计（ORM） 5](#_Toc50899938)

[1.3 Web架构设计 6](#_Toc50899939)

[1.4 项目目录结构（持续更新） 6](#_Toc50899940)

[1.4.1 NLP\_Vue前端目录结构(部分) 6](#_Toc50899941)

[1.4.2 NLP\_Flask后端目录结构（部分） 7](#_Toc50899942)

[1.5 任务类型划分 7](#_Toc50899943)

[2 团队协作指南 8](#_Toc50899944)

[2.1 分支维护 8](#_Toc50899945)

[2.2 成员开发步骤 8](#_Toc50899946)

[3 前端需求 9](#_Toc50899947)

[3.1 登录/注册 9](#_Toc50899948)

[3.1.1 登录 9](#_Toc50899949)

[3.1.2 注册 9](#_Toc50899950)

[4 后端API与需求 10](#_Toc50899951)

[4.1 登录/注册 10](#_Toc50899952)

[4.1.1 登录 10](#_Toc50899953)

[4.1.2 注册 10](#_Toc50899954)

# 1总体设计

## 1.1 模块划分

整个项目展示给用户的功能模块，核心部分分为过程管理（图 1）（一期工程完成）、数据管理（图 2）（一期工程完成）、图形界面（二期工程完成）。三期工程预计完善使用指南、社区交流模块、管理员模块、个人信息模块、团队模块。

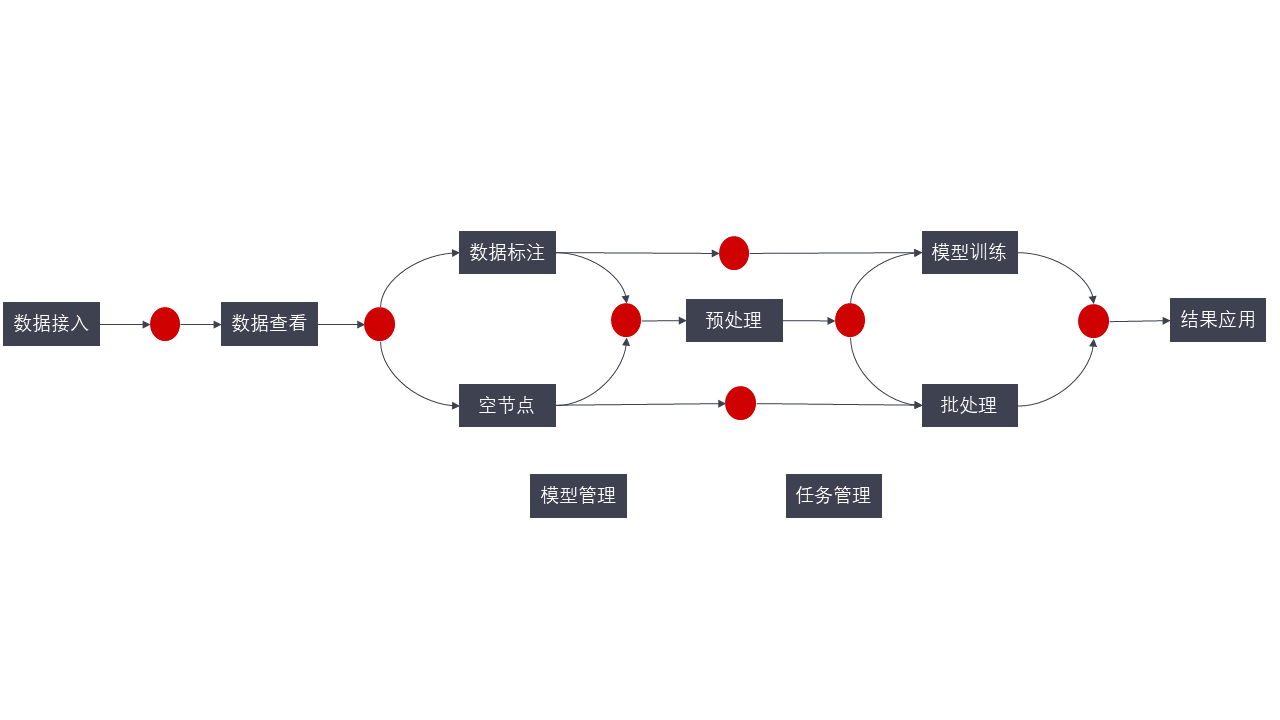


图 1 过程管理

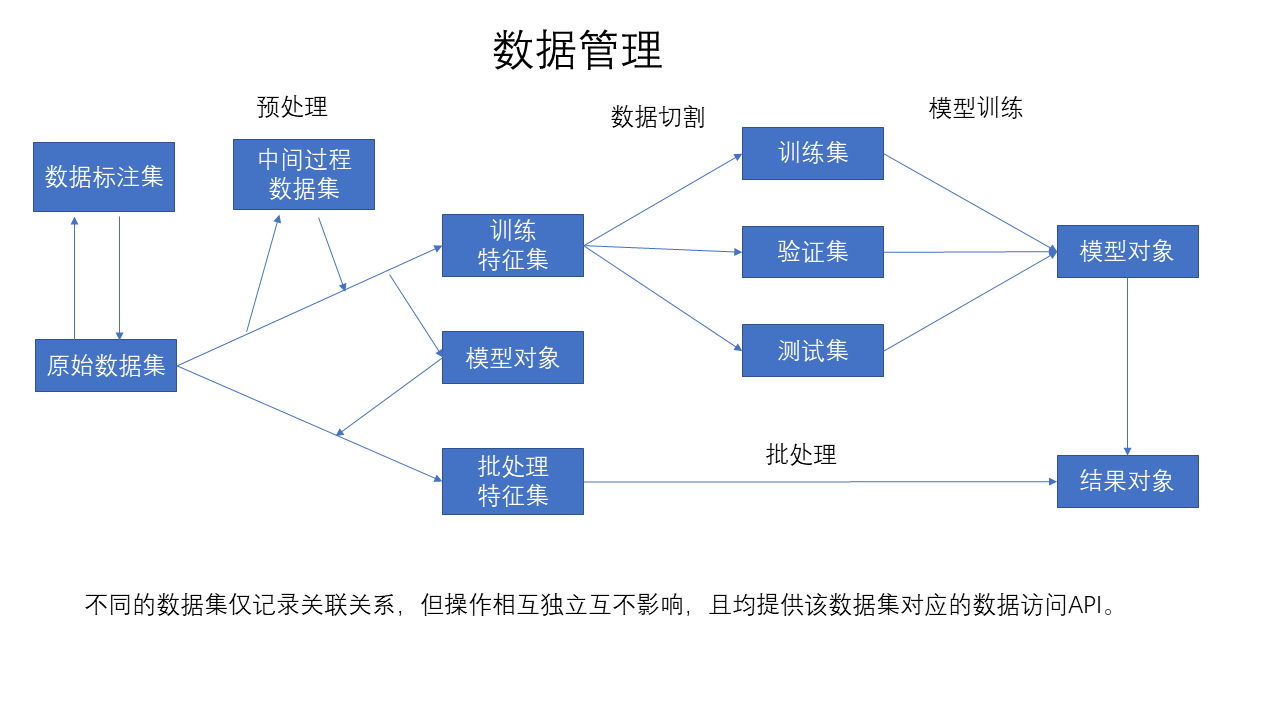


图 2 数据管理

模块跳转要求：1.对于任意过程中的数据集，能够跳转至该数据集的过程管理页面，查看该过程在NLP全流程中所处的位置和各步骤的实现情况。2.对于任意数据集，能够跳转至该数据集的数据管理模块，无限追溯前后文。

### 1.1.1 过程管理模块

该模块下辖9大功能，功能简要描述如下：

（1）数据接入：可接入训练数据集或批处理数据集，批处理数据集支持批量接入；数据接入方式完成文件上传。（预计二期工程完成数据库接入、分布式数据引擎接入、接口数据接入）。

（2）数据查看：分为训练数据集和批处理数据集；提供CRUD，提供按标签快速整理数据功能，提供常用数据清洗方法。（预计二期工程完成提供数据连接、数据追加、数据差集、数据拆分、数据分解等数据融合方法。）

（3）数据标注：提供在线标注功能；允许多用户标注。（预计二期工程提供常用标注算法辅助，允许用户上传辅助标注算法）。

（4）预处理：提供常用预处理、特征工程、特征选择方法。生成预处理算子对象，纳入模型管理模块统一管理。提供CRUD，数据访问API。允许自定义特征代码编写，代码纳入模型管理模块统一管理。提供Spark模式和Spark API。

（5）模型训练：可视化界面下配置数据集，完成数据切割。提供常用模型，纳入模型管理模块统一管理。允许自定义模型代码编写，代码纳入模型管理模块统一管理。提供Spark模式和Spark API。

（6）批处理：允许使用模型管理模块中拥有权限的模型进行处理。

（7）结果应用：提供结果查看和数据访问API。提供该结果的常用评价函数。提供下游NLP任务拓展（一至三期工程逐步添加）。（二期工程预计完成模型快速打包功能）。

（8）模型管理：管理系统提供或用户上传的预处理模型、批处理模型。

（9）任务管理：管理每一个执行中、已执行的任务并提供跳转。

### 1.1.2 数据管理

对每一个处于关系网络的数据集，不同的数据集仅记录关联关系，但操作相互独立互不影响，且均提供代码编写模块和该数据集对应的数据访问API。

### 1.1.3 图形界面

独立模块，二期工程完成。

## 1.2 数据库设计（ORM）

试行设计图如图 3所示。在本文档后续版本迭代中完善细节。

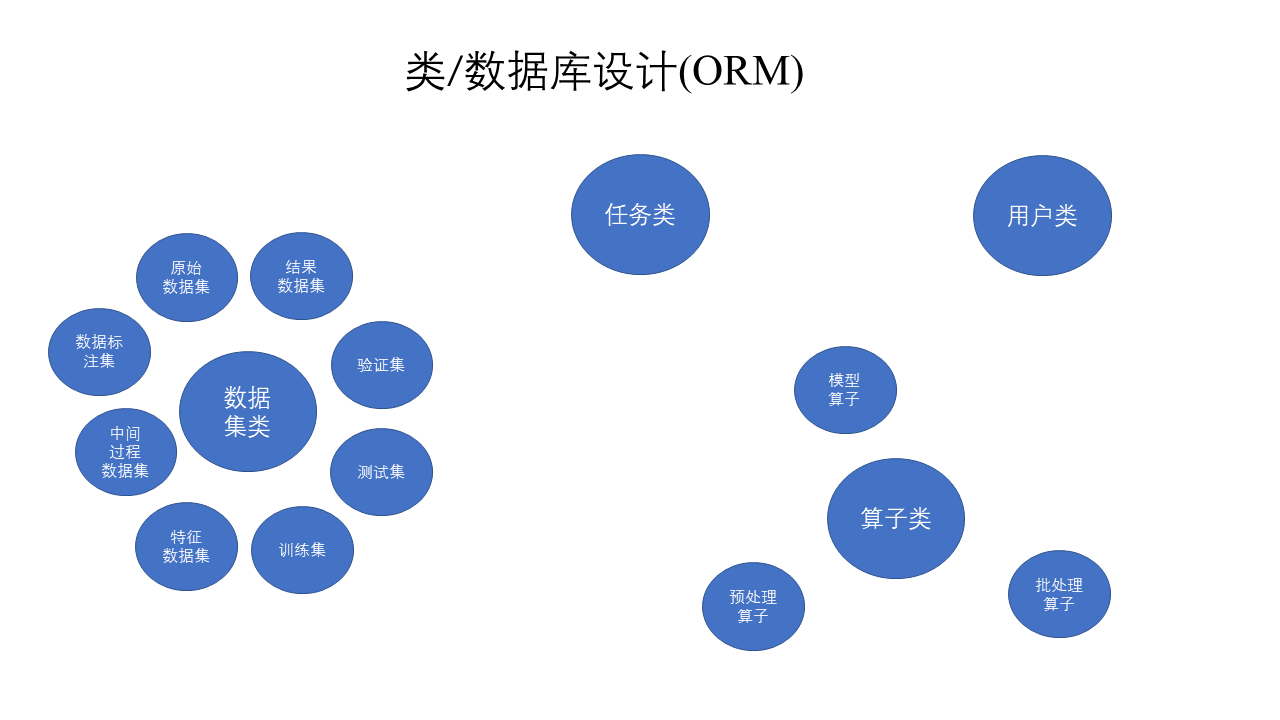


图 3 数据库设计

## 1.3 Web架构设计

试行web架构设计如图 4所示。

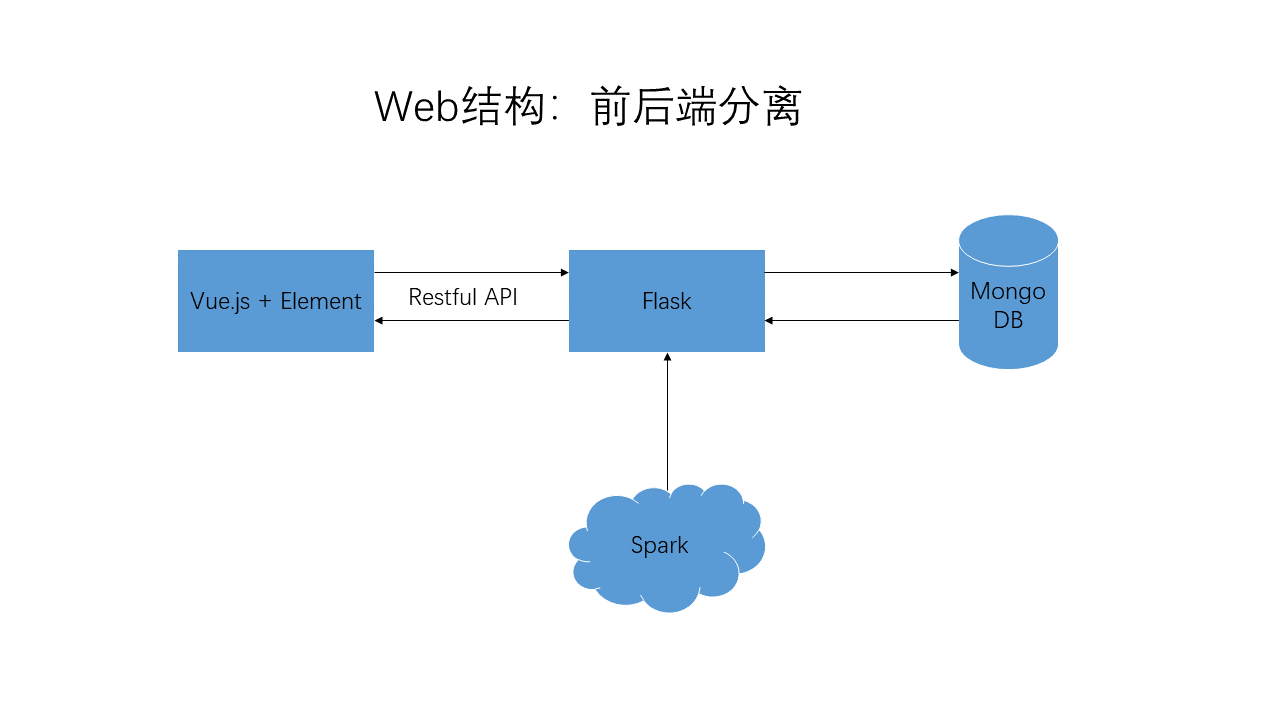


图 4 web架构

## 1.4 项目目录结构（持续更新）

### 1.4.1 NLP\_Vue前端目录结构(部分)

- src/views

- admin 管理员页面

- community 社区交流页面

- data-manage 数据管理页面

- error-log 错误日志页面

- error-page 404页面

- gui 图形化界面

- guide 使用指南页面

- home 系统主页

- login 登录页面

- process-manage 过程管理页面

### 1.4.2 NLP\_Flask后端目录结构（部分）

- app

- libs 外部库

- process\_manage\_api 过程管理API

- service\_api 服务类API

- utils 工具类

- Spark Spark引擎类

- NLPModels NLP模型类

## 1.5 任务类型划分

具体支持的任务类型包括：

1). 通用单文本分类：输入为文档内容（单文本），输出为文档类型，可以是单标签，也可以是多标签。

2). 情感分析/意图识别：输入为文档内容（文本+方面词典），输出为方面及极性标签。

3). 实体关系抽取：输入为文档内容，输出为实体及实体之间关系。

4). 文本关系分析：输入为文档内容（每一个文档为双文本），输出为关系类型。

5). 文本摘要：输入为文档内容（单个长文本），输出为摘要文本。

6). 文本排序学习：输入为文档内容（查询文本+结果文本列表），输出为排序结果。

# 2 团队协作指南

## 2.1 分支维护

本项目在一期工程开发过程中维护4个分支：

master分支。对项目进行tag或发布版本等操作，必须在该分支上进行。

develop分支，从master分支上检出。团队成员不会直接更改该分支，而是分别从该分支检出自己的feature分支，开发完成后将feature分支上的改动merge回develop分支。

feature分支，从develop分支上检出。团队成员中每个人都维护一个自己的feature分支，并进行开发工作，开发完成后将此分支merge回develop分支。

fix分支，由develop分支检出，用作bug修复，bug修复完成需merge回develop分支，并将其删除。该分支属于临时性分支。

## 2.2 成员开发步骤

以前端项目NLP\_Vue为例

（1）拉取项目至本地

git clone <https://github.com/NLPPlat/NLP_Vue.git>

（2）切换至develop分支

git checkout develop

（3）从develop分支新建并检出自己的feature分支

git checkout -b feature-xx develop

其中，xx为自己的名字。

（4）进行开发（开发前注意安装相关依赖）并使用git add git commit等命令提交改动

（5）切换回develop分支

git checkout develop

（6）更新远端代码，查看develop分支是否有更新

git pull origin develop

（7）切换回feature分支

git checkout feature-xx

（8）合并develop分支到feature分支，若有冲突需解决冲突，可联系我来查看

git rebase develop

（9）切换回develop分支

git checkout develop

（10）合并feature分支到develop分支

git merge --no-ff feature-xx

（11）推送develop分支到远端

git push origin develop

后续继续进行开发操作时，从目前的develop分支检出自己的feature分支即可。

# 3 前端需求

## 3.1 登录/注册

### 3.1.1 登录

普通登录已完成，略。

社交帐户登录在三期工程中完成。

### 3.1.2 注册

（1）点击登陆页面“没有账号？”，进入新的注册页面或弹出注册悬浮框；

（2）设计表单，需包含用户名、密码、密码二次确认、姓名（选填）、头像（选填）、个人介绍（选填）；

（3）表单失去焦点时对表单合法性进行确认，用户名不小于2字符，密码不小于8字符。

# 4 后端API与需求

## 4.1 登录/注册

### 4.1.1 登录

（1）登录

URL：/service/token

Method: POST

请求参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 类型 | 定义 | 备注 |
| username |  | 用户名 |  |
| password |  | 密码 |  |

（2）获取用户信息

URL：/service/user

Method: GET

请求头： ‘ bearer ’+token

请求参数：无

### 4.1.2 注册

（1）注册

URL：/service/user

Method: POST

请求参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 类型 | 定义 | 备注 |
| username |  | 用户名 |  |
| password |  | 密码 |  |
| name |  | 姓名 | 选填 |
| avatar |  | 头像 | 选填 |
| introduction |  | 个人介绍 | 选填 |