技术选型文档

古诗人时空流动性分析系统

# 系统技术部分划分

古诗人时空流动性分析系统整体可以分为两部分：

* OLAM系统可视化：  
  通过对原始数据构建三个维度的数据立方体，并在其上进行数据挖掘，将挖掘结果进行可视化的前端展示。前端展示部分细分可以包含有：
  + 人生轨迹可视化增强版，包含更多的信息以及更强的搜索能力
  + 诗人人生轨迹OLAM可视化，基于对数据维度的上卷、下钻、切片操作，实时展示不同数据范围内的可视化范围结果
  + 流动性分析，基于轨迹嵌入方法，通过学习嵌入和选择不同的聚类算法内核，实时可视化展示基于嵌入方法和聚类方法的流动性分析结果
  + 整体时空流动性可视化，以地球或地图数据可视化的方式，动态展示随不同年份的变化，诗人的人生轨迹迁徙的时空可视化
* 文档性质介绍网站：
  + 包含对于系统目的的说明
  + 包含对于系统分析成果的解释
  + 包含对于系统的使用说明介绍

# 时空流动性分析系统

## 前端选型

鉴于Vue3相关库的生态截至到2021年5月6日，仍然处于大量的beta阶段，所以本系统的开发仍然是基于Vue2的相关生态。

* Nuxt.js（Vue2 Based）
* Vuetify.js
* Nuxt/Axios
* Nuxt/i18n
* Amap Vue
* ECharts.js
* 不使用 prettier
* 不适用TypeScript（由于是基于Vue2）

### 开发变更

见第5节。

## 后端选型

后端选型可以分为三部分，API后端、数据预处理后端以及数据库

* API后端，主要提供RESTful API，选型 spring boot与MyBatis+Swagger+Lombok，打包jar时使用混淆等加密特性
  + 总结Yaml博客和swagger博客
* 数据预处理后端，用于预先处理需要处理的数据，使用 python 生态
* 数据库后端，使用 mysql 按特定方法存储数据立方体

# 系统成果与介绍文档

本部分为static站点，其主要目的有三：系统目的介绍、系统分析结果介绍、系统使用说明。其中系统目的介绍、系统分析结果介绍可以从论文中摘取。

* 使用VuePress（Vue2 Based）进行生成
* +Index Page
* +Search Bar
* +Multi Language
  + 可以总结博客

# 容器化部署

对格来格来诗词大数据项目，使用docker进行部署，需要处理的部分包含但不限于：

* MySQL数据库
* Django项目
* 本科毕设后端flask
* 本科毕设前端
* 流动性系统前端
* 流动性系统后端
* 静态站点

使用docker file打包，测试环境使用WSL2（On XPS13）进行。

# 实际最终选型

本节之前部分内容均为在项目早期规划时编写。

本节在系统开发完成后编写，已经与早期规划时产生了大量的分歧和差异。

## 前端选型

最终，为了更好的引入TypeScript，便于整个系统的开发工程化规范，前端选型敲定基于Vue 3的生态系统。在技术选型敲定并开始编码时，纳入考虑的UI库只有Naïve UI一家已发布release版本。为了更好使用Bootstrap风格的响应式布局来适配移动端，Element Plus UI库（即便只是beta版本）也会纳入使用（只是用其布局组件）。

同样的，兼容Vue 3的Vue Cli和Vuex工具也被纳入使用（并兼容做TypeScript支持）。

为了能够提供更好的中英文国际化体验，beta版本的vue-i18n也被使用。

常见的图表库老朋友ECharts已更新至TypeScript友好的V5版本，自然纳入考虑并使用。

高德地图通过官方AMap JS Loader引入，使用AMap基础组件以及Loca数据可视化。

Prettier和ESLint工具依旧被使用。

具体的前端依赖以及版本信息可见package.json以及package-lock.json文件。

## 后端选型

后端选型采用最为普遍的Spring Boot架构，基于Java 11开发，使用Spring Boot的传统SpringMVC，引入MyBatis连接MySQL v8数据库，采用LomBok简化开发。同时后端使用基于Swagger-UI的API文档方式。配置文件分为开发和生产两个配置。

由于SpringBoot项目的混淆和加密无法对全项目所有业务代码进行，故取消。

数据预处理使用Python生态工具。

MySQL以事实星座的方式存储数据。

具体的后端依赖以及版本信息可见pom.xml文件。

## 文档系统

原计划基于VuePress制作的使用说明文档网站因工作量问题以及时间问题被砍掉。

## 部署方案

原计划基于docker的容器化部署取消，直接使用传统的部署方式（前端Nginx，后端Java）。