**智能创作平台**

**概要设计文档**

|  |  |
| --- | --- |
| 修订时间 | 2022年7月6日 |

目 录

[目 录 1](#_Toc108033286)

[一、 引言 1](#_Toc108033287)

[1.1 编写目的 1](#_Toc108033288)

[1.2 范围 1](#_Toc108033289)

[1.3 术语和缩略词 2](#_Toc108033290)

[1.4 参考资料 2](#_Toc108033291)

[二、 体系结构设计 2](#_Toc108033292)

[2.1 系统架构设计 2](#_Toc108033293)

[2.2 需求复审 3](#_Toc108033294)

[2.3 功能模块设计 4](#_Toc108033295)

[三、 接口设计 6](#_Toc108033296)

[四、 数据设计 7](#_Toc108033297)

[4.1 用户数据库 7](#_Toc108033298)

[4.2 文章数据库 7](#_Toc108033299)

[4.3 其他数据库 8](#_Toc108033300)

[五、 测试部分 8](#_Toc108033301)

# 引言

## 编写目的

本说明的目的是对智能创作平台进行概要的功能说明，以便用户及项目开发人员了解产品大致的设计与实现。以下叙述将结合文字描述、数据流图、变换图等来描述智能创作平台的体系结构设计、接口设计，数据设计等方面。本说明的预期读者有客户、项目经理、开发人员以及跟该项目相关的其他竞争人员。

## 范围

### 系统目标

本软件平台旨在实现基于自动文本摘要的智能创作品平台系统，为新闻工作、舆论分析人员自动提供新闻文本的标题、摘要、关键词，提高创作水平，为新媒体AI赋能加速。如何从海量文本信息中快速而准确的获取人们所需要的内容，成为了当今社会上一种非常迫切的需求。为这些不规范的文本信息生成一条简洁、切合原文表达的标题可以缓解信息过载压力，提高工作效率并为相关部门监察社会舆论提供帮助。

### 主要软件需求

#### 功能需求

* 用户注册与登录：获取用户输入的用户名、密码进行登录和注册，并可以修改自己的密码
* 文章管理：通过文章标题或标签进行模糊搜索显示结果列表、创建新文章、删除自己的文章、修改自己创建的文章
* 标题、摘要、关键词生成：使用人工智能算法对给出的文本进行标题、摘要、关键词生成
* 数据大屏可视化展示（可交互图表、词云、爬虫榜单等）：为用户创作提供相关数据和信息，便于复盘总结写作情况。

#### 非功能需求：

* 数据需求：用户信息、文章信息等需要进行安全可靠的数据存储。
* 性能需求：对于响应时间、更新处理时间和数据的转换及传送时间有相应的时间限制。
* 运行需求：用户界面在用户操作后发出相应的响应。软件接口和硬件接口提供限制。

### 软件设计约束、限制

采用免费的教育版软件Pycharm，Web开发选择Django开发框架，人工智能算法采用Tensorflow框架，开发成本低，但运行效率可能受限。技术上，采用现有的小组成员掌握的技术保证项目的最终实现。环境上，在本机保证运行正确。

## 术语和缩略词

* 用户名：使用此系统的用户登录时所使用的字符串，用户的用户名唯一，一个用户对应一个用户名和一个自动生成的用户ID。
* 关键词：与文章内容有关的一系列词组的集合，突出文章重点。
* 摘要：与文章内容有关的一系列语句的集合。能简单概括整篇文章的主要内容，突出文章重点。
* 标题：与文章内容有关的一句话，简洁明了地表明文章的中心论点。

## 参考资料

* 可行性研究报告
* 工程开发方案
* 需求规格说明书
* 概要设计说明书
* 详细设计说明书
* 测试方案
* 《软件工程方法与实践 (第3版)》, 窦万峰等编著, 机械工业出版社，2016

# 体系结构设计

## 系统架构设计

系统旨在实现基于自动文本摘要的智能创作品平台系统，为上层人员直接提供新闻文本的标题、摘要、关键词。系统使用TextRank算法对原始新闻数据做抽取预处理，系统的核心模型利用了流行的深度学习方法，基于多端注意力机制构建了Bert+UniLM算法。并将算法模型部署在服务器端，由Web界面直接调用后台接口，用户人员可以通过门户网站方便使用。智能创作平台系统由数据处理层、算法模型层和应用服务层三部分组成，系统总体架构如图 2.1‑1所示。

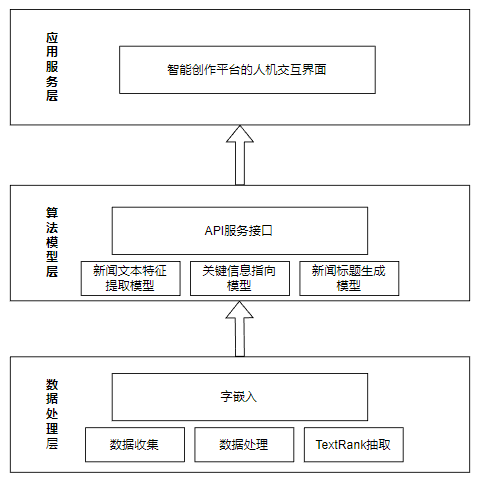


图 2.1‑1 智能创作平台系统总体架构图

整个系统架构自下往顶的结构共三层：

**数据处理层**：位于系统架构的最下面一层，是整个系统的基础。数据处理层包括数据收集、数据处理和使用TextRank算法对长文本类型的党建新闻预抽取三部分，主要负责新闻数据的预处理功能，包括中文分词，过滤标点符号、特殊符号，去除常见词、停用词等，是连接算法模型层的输入接口。

**算法模型层**：位于中间层，是整个系统的核心模块。算法模型层的主要功能是自动文本摘要算法的选择与实现，包括新闻文本特征提取模块、关键信息指向模块和新闻标题生成模块三部分。算法模型层根据组合算法的功能又可以看作是三个阶段，即新闻文本特征阶段、关键信息重组阶段和新闻标题、摘要生成阶段。

**应用服务层**：位于最上一层，是人与系统算法模型进行交互的界面。算法模型经过党建新闻数据的多轮训练后，使用Django部署在服务器端。为了增强人机交互体验，设计门户网站界面，通过Django传递的API接口调用服务器端的模型。人们只需要在系统门户网站的文本框输入长文本类型的党建新闻，就可以得到算法模型生成的新闻标题。

## 需求复审

### 顶层数据流图

根据系统的需求分析用例图，画出顶层数据流图如图 2.2‑1所示：

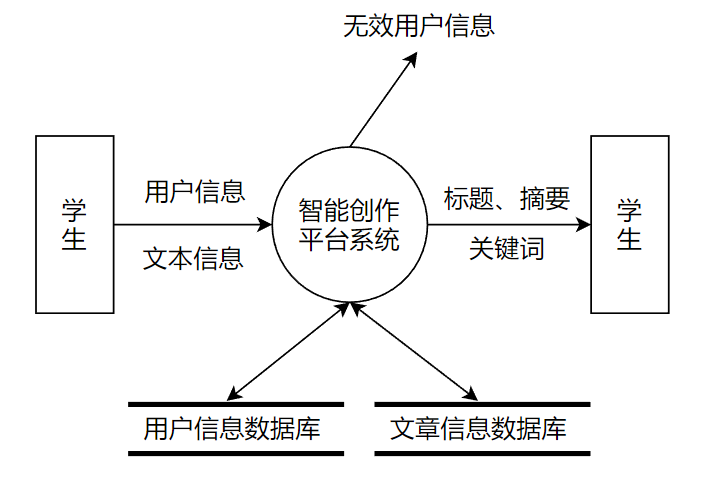


图 2.2‑1 智能创作平台顶层数据流图

### 第二层数据流图

根据顶层数据流图，做出第二层数据流图如图 2.2‑2所示：

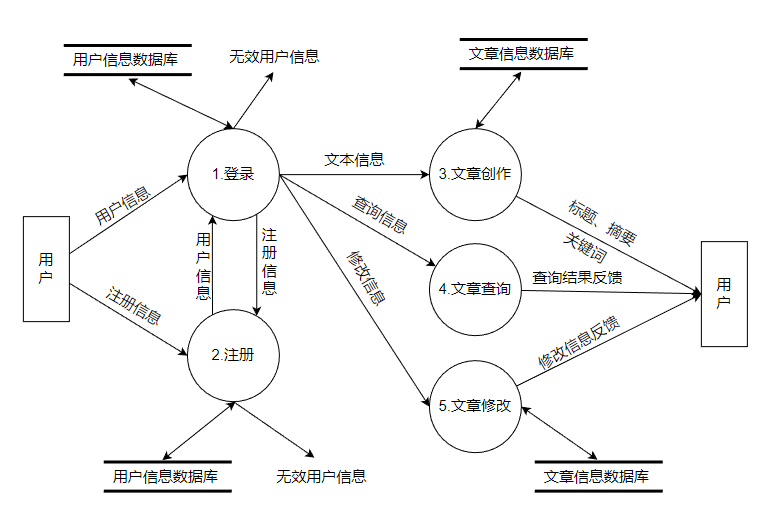


图 2.2‑2 智能创作系统第二层数据流图

### 第三层数据流图

根据第二层数据流图，画出第三层数据流图如下（以登录与文章创作为例）：

用户登录第三层数据流图如图 2.2‑3所示：

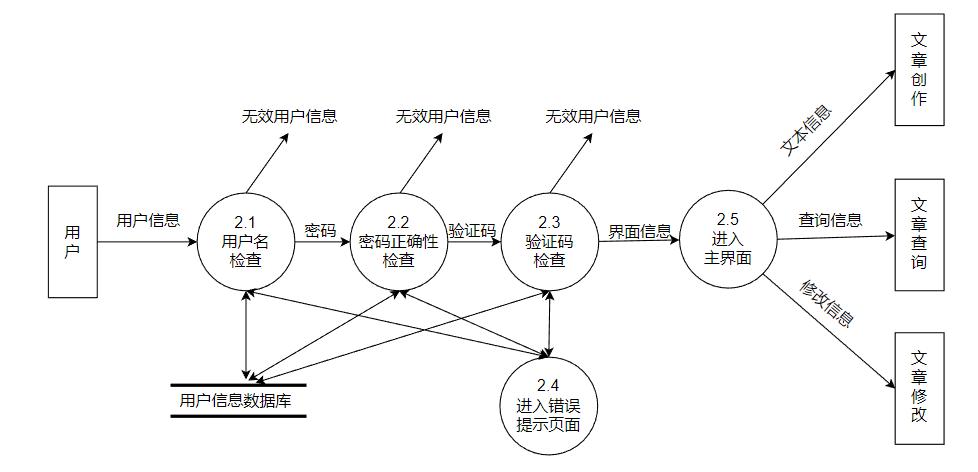


图 2.2‑3 用户登录第三层数据流图

文章创作第三层数据流图如图 2.2‑4所示：

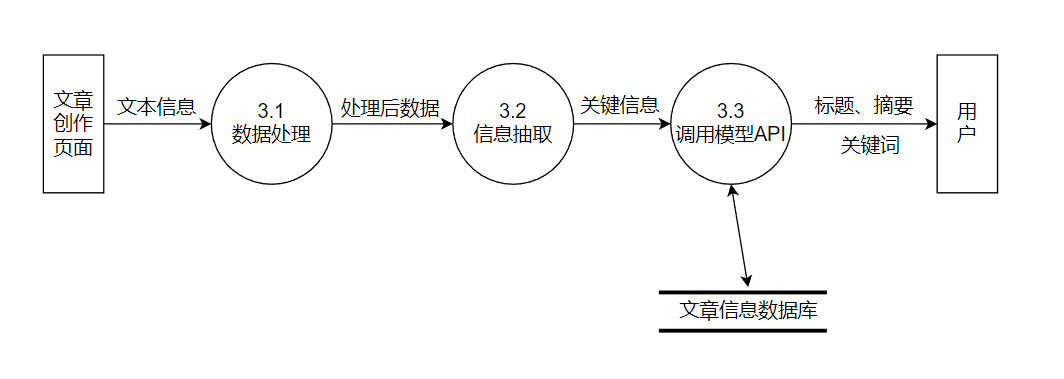


图 2.2‑4 文章创作第三层数据流图

## 系统非功能性需求分析

智能创作品平台系统在必要的功能性需求基础上，还需要考虑到非功能性需求。非功能性需求是保证系统能够平稳运行的一些因素分析，主要包括实用性需求分析、高内聚低耦合性需求分析、易用性需求分析、可扩充性需求分析和系统性能需求分析等。智能创作品平台系统主要的非功能性需求分析具体如下。

### 实用性需求分析

利用深度学习技术构建算法模型并搭建用户Web平台，为用户提供新闻标题自动生成功能，使得人们能够通过移动终端设备快速的、简洁的、随时随地的获取文本中的主要内容。

### 高内聚低耦合性需求分析

内聚性是指系统内部功能关系聚集、关联的程度，耦合性是指系统结构中各个功能模块之间相互联系的紧密程度。如果不同功能模块之间联系紧密，那么在提出新的系统功能需求时就会出现“牵一发而动全身”的局面，一处更改，处处更改，导致系统代码的可维护性特别差。本文设计的智能创作品平台系统各个功能模块层相对独立，每个功能模块负责相对独立的功能，功能模块内部元素之间的联系保持紧密结合。功能模块之间通过接口交互，而不依赖其他功能模块。系统的高内聚低耦合性有效提高了代码的重复利用率。

### 易用性需求分析

易用性需求分析要求系统操作简单，用户无需参考复杂的文档，通过系统网站界面即可实现所需要的任务功能。本系统的开发过程中遵循高内聚低耦合原则，用户无需操作后端的算法模型源码，只在前端页面进行操作就可以完成系统功能。系统网站界面的模块化，输入框、输出框和操作按钮简单明了，十分符合人机交互原理。

### 可扩充性需求分析

可扩充性指当提出新的功能需求时，系统扩充功能简易方便。本系统开发过程中遵循高内聚低耦合原则，各功能模块分别根据其独立的功能创建接口。在扩充新功能时，只需在相应的功能模块实现后定义好接口供其他模块调用，在扩充过程中不会影响到其他功能模块的正常运行。

### 系统性能需求分析

系统性能性需求是指系统能够满足用户任务需求的同时，还能够稳定、高效的保持正常运行。用户正常使用系统的功能时，系统能够及时做出反馈。

## 功能模块设计

根据党建新闻标题生成系统的总体架构分析可知，系统由数据处理层、算法模型层和应用服务层三部分组成。数据处理层收集并清理数据，做模型输入前的一些预处理工作；算法模型层对数据处理层的输入使用多端注意力机制做特征提取，使用指针网络指向新闻文本中的重要信息，融入到生成的新闻摘要中，保证系统生成流畅、精简的新闻标题，是整个系统设计的核心模块；应用服务层将训练好的模型部署在设计好的Web门户网站，方便人们直接网页操作。系统详细功能设计图如图 2.3‑1所示。

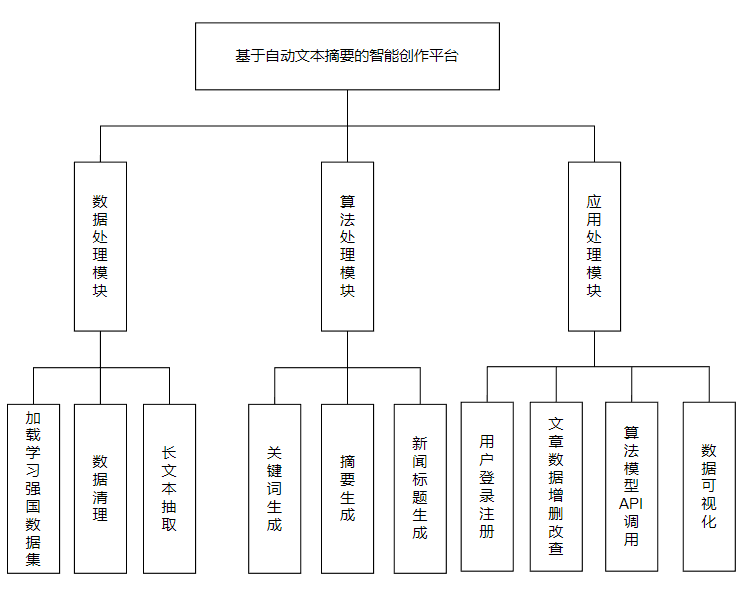


图 2.3‑1 智能创作系统详细功能设计图

系统流程图如图 2.3‑2所示。

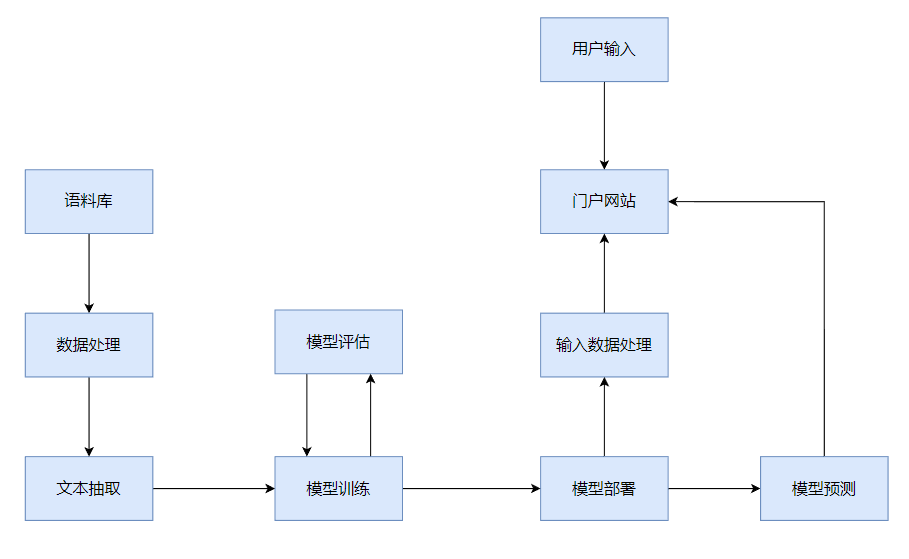


图 2.3‑2 智能创作系统系统流程图

# 接口设计

## 用户接口

用户登录界面：当用户访问正确的网址时，进入用户登录界面。该界面至少包含用户名输入栏和密码输入栏以便接受用户的输入，还应该有“登录”按钮使得用户可以提交用户名和密码和“注册”按钮。

注册界面：用户点击用户登录界面的“注册”按钮后可以进入注册界面。该界面应提供“用户名”“密码”“确认密码”“邮箱”“验证码”栏供用户输入，还应该提供“提交”按钮让用户提交用户信息并完成注册。

主界面：用户提供正确的用户名和密码后，跳转到该界面。用户主页是由多个Layui卡片构成的，每一个卡片显示一定的数据，以创作量卡片展示了全站和用户近一周的文章新建次数，热点榜单则是使用爬虫获得的不同网站近期的热点消息或者最新消息。

文章总览界面：可通过读取主界面的搜索关键字在文章库中匹配相关文章，并将相关结果显示到该界面。

创建文章界面：可以创建新文章，可以使用AI生成功能对正文获取标题、摘要、关键词。

编辑文章界面：可对已创建文章再次更新编辑。

个人中心界面：可以用户修改密码。

使用指南和关于界面：介绍使用方法和本组成员。

# 数据设计

## 用户数据库

Auth模块是Django自带的用户认证模块：我们在开发一个网站的时候，无可避免的需要设计实现网站的用户系统。此时我们需要实现包括用户注册、用户登录、用户认证、注销、修改密码等功能。它内置了强大的用户认证系统——auth，它默认使用auth\_user表来存储用户数据。auth\_user表的设计如表 4.1‑1所示。

表 4.1‑1 用户数据库auth\_user设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段列表 | 字段描述 | 主键 | 类型 | 长度 | 说明 |
| id | id | 是 | int | 11 | 非空 |
| password | 密码 |  | varchar | 126 | 非空 |
| last\_login | 最后登录时间 |  | datetime | 6 |  |
| is\_superuser | 超级管理权限 |  | tinyint | 1 | 非空 |
| username | 用户名 |  | varchar | 150 | 非空 |
| first\_name | 名 |  | varchar | 150 | 非空 |
| last\_name | 姓 |  | varchar | 150 | 非空 |
| email | 邮箱 |  | varchar | 254 | 非空 |
| is\_staff | 管理员权限 |  | tinyint | 1 | 非空 |
| is\_active | 登陆权限 |  | tinyint | 1 | 非空 |
| date\_joined | 注册时间 |  | datetime | 6 | 非空 |

## 文章数据库

Django可以根据自身需要，创建新的数据库，例如在创建writing应用后，需要一个存储文章数据的数据库，在相应model文件中编写Article类，然后使用数据库迁移命令，即可创建相应的数据库writing\_arcitle，数据库还支持多次更新多次迁移，这为拓展应用功能，后期产品迭代提供了很好的基础。我们的文章数据库设计如表 4.2‑1所示。

表 4.2‑1 文章数据库writing\_arcitle设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段列表 | 字段描述 | 主键 | 类型 | 长度 | 说明 |
| id | id | 是 | bigint | 20 | 非空 |
| title | 文章名 |  | varchar | 50 | 非空 |
| abstract | 摘要 |  | longtext | 0 | 非空 |
| body | 正文 |  | longtext | 0 | 非空 |
| author | 作者 |  | varchar | 10 | 非空 |
| create\_time | 创建时间 |  | datetime | 6 | 非空 |
| update\_time | 最后更新时间 |  | datetime | 6 | 非空 |
| category | 分类 |  | varchar | 5 | 非空 |
| keyword | 关键词 |  | varchar | 40 | 非空 |

## 其他数据库

Django的Auth模块为我们提供了其他的与用户认证、数据库连接有关的数据库，如和session数据存储有关的django\_session表、和数据库迁移更新记录有关的django\_migrations表、存储所有的关系表的django\_content\_type表，但因为是模块自带的数据库，并不是软件主要设计内容，在此不详细展开说明。

# 测试部分

集成策略：采用自底向上的增量式集成方式

测试方针：首先对规定的正常的功能进行测试，以黑盒测试为主，白盒测试为辅助来设计测试用例，测试用例要尽量覆盖所有的程序功能，记录测试中的问题。