**南开大学Web搜索引擎系统说明文档**

**一、📖 系统概述**

**项目背景**

南开大学Web搜索引擎是专门针对南开大学校内资源构建的专业搜索系统，为师生提供高效、准确的校内信息查询服务。系统采用现代Web技术栈，实现了从网页抓取到个性化推荐的完整搜索解决方案。

**核心目标**

* 🎯 **专业性**：专注南开大学校内资源
* 🚀 **高效性**：快速准确的搜索响应
* 👤 **个性化**：基于用户行为的个性化推荐
* 📊 **智能化**：多种高级搜索功能

****

* 项目爬取了南开新闻、学术资源、教务、校史、国际合作、校友会、学工会、高等教育研究所、招生办 以及各学院的数据总计10万+，筛选后每个网页保存为独立的JSON文件，在10w+数据上用Whoosh库在多个索引域上构建索引。

|  |
| --- |
| [      "https://news.nankai.edu.cn", "https://jwc.nankai.edu.cn",      "https://www.nankai.edu.cn", "https://zsb.nankai.edu.cn",      "https://lib.nankai.edu.cn","http://international.nankai.edu.cn",      "https://xgb.nankai.edu.cn","https://law.nankai.edu.cn",      "http://less.nankai.edu.cn","https://xs.nankai.edu.cn/",      "https://cc.nankai.edu.cn","https://cs.nankai.edu.cn",      "https://ai.nankai.edu.cn","https://ceo.nankai.edu.cn",      "https://finance.nankai.edu.cn","https://math.nankai.edu.cn",      "https://physics.nankai.edu.cn","https://chem.nankai.edu.cn",      "https://economics.nankai.edu.cn","https://bs.nankai.edu.cn"  ]  # 在custom\_analyzers.py中定义的Schema  schema = Schema(  id=STORED, # 文档唯一ID (存储)  title=TEXT(analyzer=analyzer, stored=True), # 标题 (索引+存储)  content=TEXT(analyzer=analyzer, stored=True), # 内容 (索引+存储)  url=STORED, # URL (仅存储)  publish\_date=DATETIME(stored=True), # 发布日期  crawl\_time=DATETIME(stored=True), # 抓取时间  source=STORED, # 来源网站  filetype=STORED, # 文件类型  filename=STORED # 文件名  ) |

* 实现了站内查询和文档、短语、通配、网页快照、日志查询五种高级查询。
* 提供用户账号系统，基于不同用户的浏览记录实现个性化查询和推荐。
* 前端采用原生HTML、CSS和JavaScript构建用户界面。
* 后端使用Python的Flask框架搭建Web服务器，Flask-Login实现用户认证，JSON与SQLite
* 存储数据， Whoosh用于全文索引，Jieba进行中文分词处理。

**二、项目概要**

**项目框架**

Web-Search-Engine-main/

├── Spider/ # 网页抓取模块

│ ├── nankai\_news\_spider.py # 南开新闻爬虫

│ └── nankai\_news\_2025\_05\_29\_00\_39\_58/ # 抓取数据

├── index/ # 索引模块

│ ├── build\_index.py # 索引构建脚本

│ ├── custom\_analyzers.py # 中文分析器

│ └── index\_dir/ # Whoosh索引文件

├── query\_service/ # 查询服务模块

│ ├── app.py # Flask主应用

│ ├── users.db # 用户数据库

│ ├── query\_log.txt # 查询日志

│ ├── click\_log.txt # 点击日志

│ └── templates/ # HTML模板

│ ├── search.html # 搜索主页

│ ├── results.html # 搜索结果页

│ ├── login.html # 用户登录页

│ ├── register.html # 用户注册页

│ ├── profile.html # 个人中心

│ ├── recommendations.html # 个性化推荐

│ ├── demo.html # 功能演示

│ ├── logs.html # 查询日志

│ ├── click\_stats.html # 点击统计

│ └── redirect\_error.html # 重定向错误页

└── requirements.txt # 依赖包清单

**三、功能实现**

**3.1 网页抓取**

**在spider目录下实现网页抓取功能**

[

    "https://news.nankai.edu.cn", "https://jwc.nankai.edu.cn",

    "https://www.nankai.edu.cn", "https://zsb.nankai.edu.cn",

    "https://lib.nankai.edu.cn","http://international.nankai.edu.cn",

    "https://xgb.nankai.edu.cn","https://law.nankai.edu.cn",

    "http://less.nankai.edu.cn","https://xs.nankai.edu.cn/",

    "https://cc.nankai.edu.cn","https://cs.nankai.edu.cn",

    "https://ai.nankai.edu.cn","https://ceo.nankai.edu.cn",

    "https://finance.nankai.edu.cn","https://math.nankai.edu.cn",

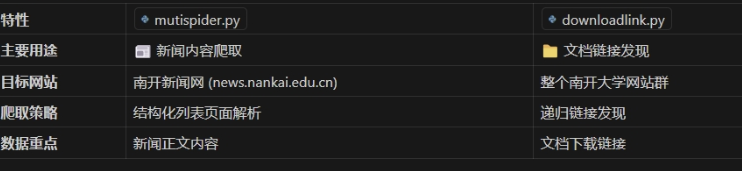
    "https://physics.nankai.edu.cn","https://chem.nankai.edu.cn",

    "https://economics.nankai.edu.cn","https://bs.nankai.edu.cn"

]

通过spider中的两个个爬虫对网页进行爬取，downloadlink（只爬取文档信息链接，用来补 充） ,mutispider.py（爬取网页和文档信息，保存网页快照）。爬取时间接近24小时。

对爬取完的数据进行数据清洗，去除一些重复的损坏的数据。









**3.2 文本索引——Whoosh**

在 index 目录下，使用ChineseAnalyzer中文分词器对爬取的内容进行分词，并构建 Whoosh索引。

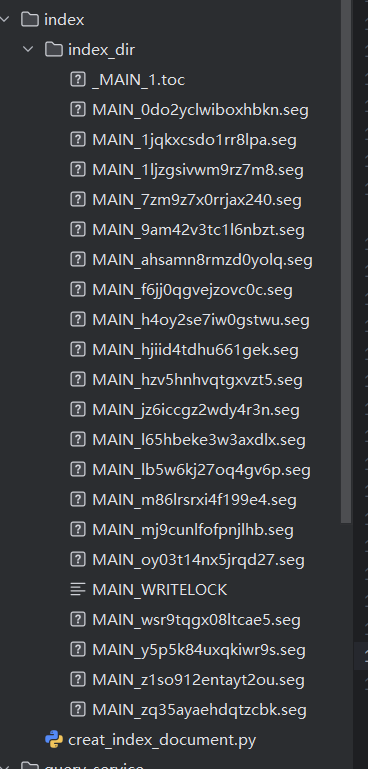
**中文分词器 (ChineseTokenizer)**

|  |
| --- |
| class ChineseTokenizer(Tokenizer):  def \_\_call\_\_(self, value, positions=True, chars=True, keeporiginal=False,  removestops=True, start\_pos=0, start\_char=0, mode='', \*\*kw):  # 使用jieba进行中文分词  seg\_list = jieba.cut\_for\_search(value)  # 为每个词创建Token对象，包含位置信息 |

**中文分析器 (ChineseAnalyzer)**

|  |
| --- |
| class ChineseAnalyzer:  def \_\_init\_\_(self):  self.tokenizer = ChineseTokenizer()    def \_\_call\_\_(self, value, \*\*kwargs):  # 返回分词结果的生成器  return (token for token in self.tokenizer(value, \*\*kwargs)) |

首先自动检测Spider目录下的数据源（包括nankai\_news\_新闻数据和download\_links\_文档数据），然后使用自定义的中文分析器（基于jieba分词）创建Whoosh索引Schema，接着分类处理不同类型的JSON数据（新闻索引完整内容、文档索引锚文本、页面索引页面内容），最后利用多进程并行写入技术将10万+文档转换为可快速检索的倒排索引文件，整个过程实现了从原始JSON数据到毫秒级搜索引擎的完整转换。



**3.4 查询服务**

**1、基础搜索：支持普通关键词搜索**

****



**普通查询处理核心代码**

|  |
| --- |
| else:  # 普通查询，使用原有逻辑  parser = MultifieldParser(["title", "content"], schema=ix.schema)  parser.add\_plugin(WildcardPlugin())  try:  parsed\_query = parser.parse(main\_query)  final\_query = parsed\_query  print(f"Debug: Main query parsed successfully: {parsed\_query}")  except Exception as e:  print(f"Debug: Query parsing error: {e}")  return render\_template('search.html', query=query\_str, error=f"Query parsing error: {e}") |

**2、文档搜索：支持PDF、DOC、DOCX、XLS、XLSX等格式**

搜索格式： 章程 filetype:pdf （关键字 +filetype：文件类型）

****

****

**3、站内查询：支持不同网站查询**

格式：活动 site:news.nankai.edu.cn （关键字+site+网址）

在news.nankai.edu.cn网址下的搜索活动的查询结果

****

**4、短语搜索：支持精确短语匹配**

格式：” 查询的短语 ”

以下是以“开学典礼”为整体的查询结果

****

**5、通配符搜索：支持\*的通配符操作**

用户可能并不知道自己要查什么，支持通配符（正则）查询操作。

例如：\*代表多个字符， “计算\*”用于查询含有计算的结果。

****

**6 、网页快照**

网页快照是搜索引擎在收录网页时，对网页进行备份，存在自己的服务器缓存里，当用

户在搜索引擎中点击链接时，搜索引擎将爬虫系统当时所抓取并保存的网页内容展现出来，

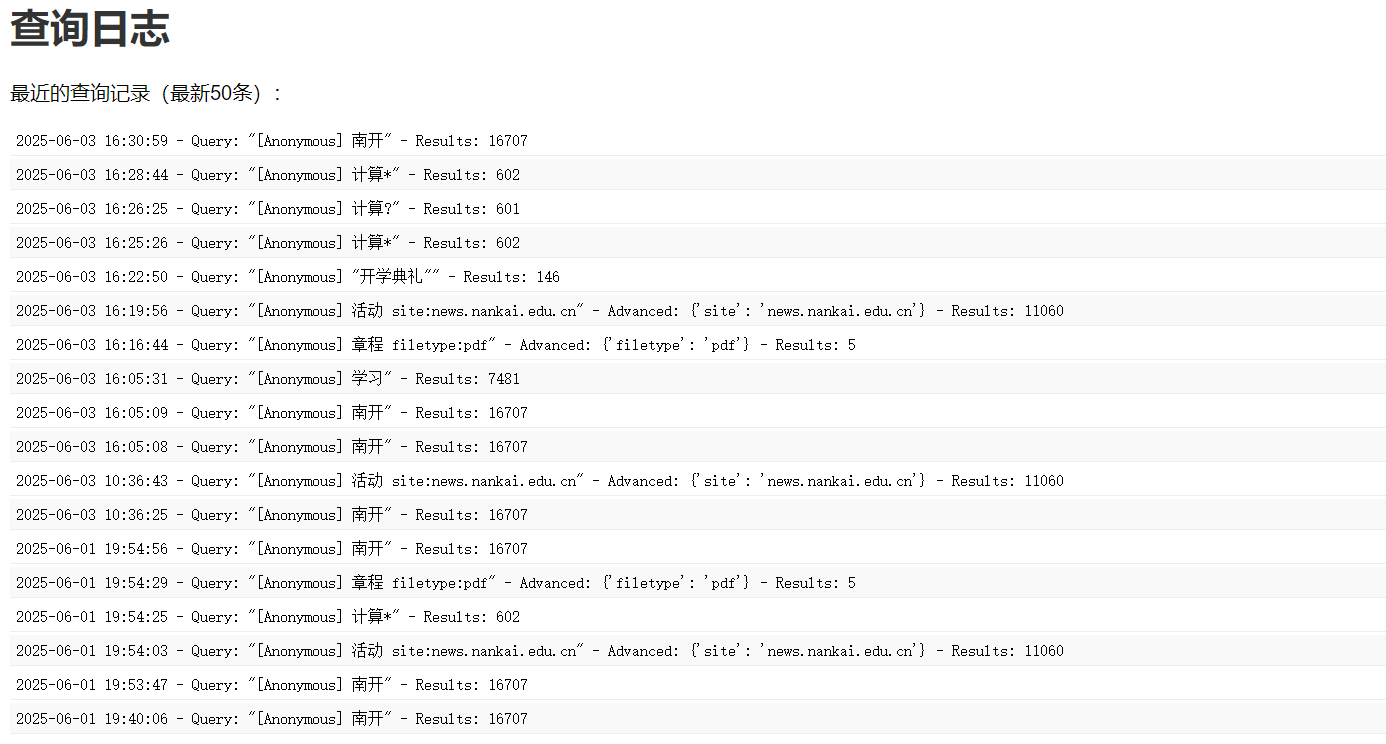
称为网页快照。





**7、查询日志**

****

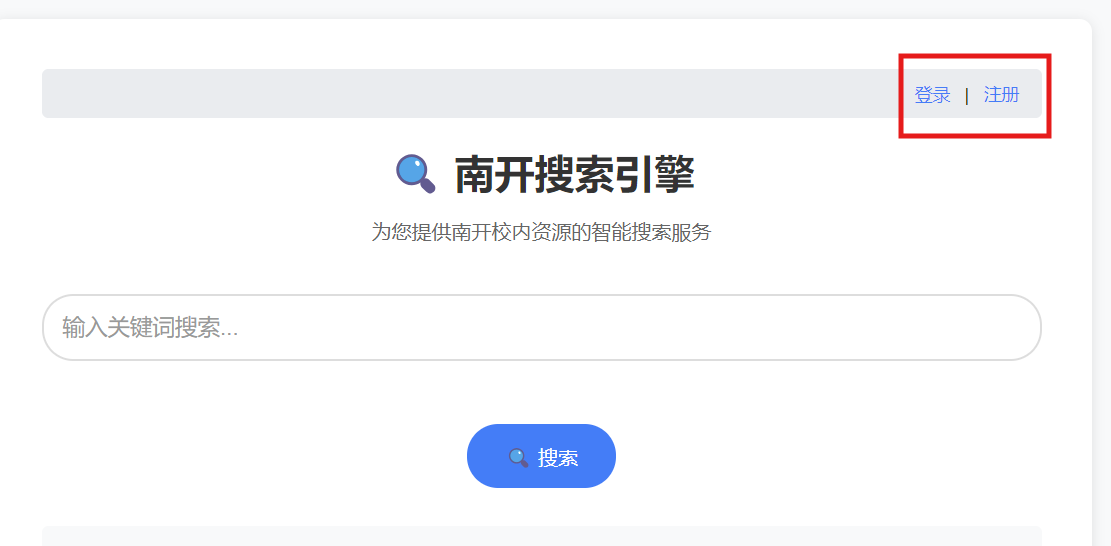
****

**包含了查询与点击分析**

****

**四、个性化查询**

**1、登录与注册系统**

****

****

****

**2、个人中心系统**

不同的用户个人中心系统不同，查询历史与查询记录也不同

****

**五、个性化推荐**

根据不同用户的查询记录与点击历史，提供不同的查询建议

