特型卡片搜索系统
设计文档

Remakers[1]

[1]容逸朗、马钿雨、蒋政、王梓桥

目录

1 概述

1.1 引言

本文档由 PRJ3B Remakers 小组编写，用于详细介绍特型卡片搜索系统的 1.0 版本。

1.2 系统概述

特型卡片搜索系统，故名思义，是面向用户的一种准确信息查找系统，由于覆盖类目和场景众多，因此要针对不同的数据给出合适的解决方向。本项目面向的正是上述的场景，由 Remakers 在 2022 年 3 月至 2022 年 5 月间开发和维护，完成了搜索页面和运营平台，以及一个支持搜索的后端服务器。北京字节跳动科技有限公司是该项目的需求方，清华大学计算机系软件工程课程为该项目提供支持。

1.3 文档概述

本文档是“Remakers 特型卡片搜索系统”的系统分析及设计文档。本文档包含了以下几部分内容:

1. 软件需求及流程介绍；
2. 软件设计架构，各模块间的关系；
3. 系统通信接口设计；
4. 系统各模块设计；
5. 数据库设计。

2 需求及分析

2.1 功能需求

2.1.1 需求详情-用户端

- 搜索句子
- 查看搜索结果

2.1.2 需求详情-管理员端

- 登录系统
  - 对用户名和密码做校验，根据用户等级决定是否开放用户管理功能
- 权限系统
  - 区分管理员、普通用户的权限
- 配置写入描述文件
- 配置用户输入句式模板
- 增加、删除、修改单一或多条结构化数据
  - 支持单一数据即时修改
  - 支持 XML 格式的文件上传
- 管理词表（意图词、垃圾词等）

2.1.3 需求详情-搜索系统

- 在线检索系统
  - 意图识别：产出本次请求的意图（要对哪些卡片进行召回）
  - query 理解：完成（若干）特定特性卡片下的槽位填充工作
  - 倒排检索：根据 query 分析结果，检索得到符合条件的 docId
  - 摘要提取：根据数据的唯一键（卡 id+docid）获取数据的展现内容
- 文件处理系统
  - 文件拆包，按需写入数据库
- 密码系统
  - 管理用户名单与密码
- 数据库系统

2.2 非功能需求

- 效率需求
  - 搜索过程的总时长不应超过 700 ms
- 便捷需求
  - 具有用户友好型的运营平台管理页面
- 美观需求
  - 搜索结果的渲染及美化
- 其他需求
  - 具有鲁棒性，登录页面能对用户不合法的输入进行检查
  - 服务器支持两百人以上的用户同时进行检索

3 体系结构设计

3.1 项目划分

按照要求，我们将项目分为以下三个部分：

- 搜索页面（用户端）
  - 搜索句子
  - 查看搜索结果
- 运营平台（管理员端）
  - 登录系统
  - 权限系统
    - 区分管理员、普通用户的权限
  - 配置写入描述文件
  - 配置用户输入句式模板
  - 增加、删除、修改单一或多条结构化数据
    - 支持单一数据即时修改
    - 支持 XML 格式的文件上传
  - 管理词表（意图词、垃圾词等）
- 后台（搜索系统）
  - 在线检索系统
    - 意图识别
    - query 理解
    - 倒排检索
    - 摘要提取
  - 文件处理系统
    - 文件拆包，按需写入数据库
  - 密码系统
    - 管理用户名单与密码
  - 数据库系统

4 通信接口设计

4.1 用户系统接口

用户管理系统和密码系统所使用的接口如下：

URL	用途
/alluser	取得所有用户名
/login	密码较验
/useradd	增加用户
/userdelete	删除用户

【GET】/alluser
用途：返回系统中所有已注册的用户名
参数格式：无参数
返回值格式：application/json

返回码	变量名称	数据类型	描述
200	data	字符串列表	用户名称列表
400	err	字符串	错误信息

【POST】/login
用途：取得对应用户的口令
参数格式：application/json

变量名称	数据类型	描述
username*	字符串	用户名称

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	password	字符串	加密后的用户密码
400	err	字符串	错误信息

【POST】/useradd

用途：新增用户

参数格式：application/json

变量名称	数据类型	描述
username*	字符串	用户名称
userpassword*	字符串	已加密的用户密码
userlevel*	字符串	用户等级

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	response	字符串	上传成功信息
400	err	字符串	错误信息

【POST】/userdelete

用途：删除指定用户

参数格式：application/json

变量名称	数据类型	描述
username*	字符串	用户名称

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	response	字符串	删除成功信息
400	err	字符串	错误信息

4.2 数据配置接口

数据辅助管理系统所使用的接口如下：

URL	用途
/category	取得所有特型卡类🄵
/pattern	数据配置文件及手动维护的词表
/setting	数据写入行为描述文件

【GET】/category
用途：返回系统中所有特型卡的名字
参数格式：无参数
返回值格式：application/json

返回码	变量名称	数据类型	描述
200	data	字符串列表	特型卡列表
400	err	字符串	错误信息

【POST】/pattern
用途：上传配置文件和词表
参数格式：application/json

变量名称	数据类型	描述
data*	字符串列表	数据内容
operation*	字符串	操作类型 (insert, delete)
resource*	字符串	特型卡名称
type*	字符串	上传信息类型 (intent, garbage, pattern)

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	message	字符串	上传成功信息
400	err	字符串	错误信息

【GET】/pattern

用途：取得对应类🄴的配置文件和词表

参数格式：url parameter

变量名称	数据类型	描述
resource*	字符串	特型卡名称
type*	字符串	提取信息类型 (intent, garbage, pattern)

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	data	字符串列表	对应数据
400	err	字符串	错误信息

【POST】/setting

用途：上传写入行为描述文件。注意上传数据中需包含 data 和 file 中的至少一个，否则也会返回 400。

参数格式：multipart/form-data

变量名称	数据类型	描述
data	字符串	JSON 格式上传之数据内容
file	文件	文件格式上传之数据内容
resource*	字符串	特型卡名称

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	message	字符串	上传成功信息
400	err	字符串	错误信息

【GET】/setting

用途：取得对应类🄴的配置文件和词表

参数格式：url parameter

变量名称	数据类型	描述
resource*	字符串	特型卡名称

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	data	字符串列表	对应数据
400	err	字符串	错误信息

4.3 数据接口

数据系统所使用的接口如下：

URL	用途
/data	数据上传、下载接口
/dataname	取得已上传文件的名字
/item	取得特定的数据
/key	取得数据的所有标签名

【POST】/data

用途：后端接收 XML 数据之接口。若为插入，则需要有 data 或 file 中的一个；若为删除，则需要 key 或 filename 中的一个。（若同时出现则两者都删除）

参数格式：multipart/form-data

变量名称	数据类型	描述
data	字符串	JSON 格式上传之数据内容
file	文件	文件格式上传之数据内容
filename	字符串	文件名
key	字符串	欲删除数据之索引值
resource*	字符串	特型卡名称
type*	字符串	操作类型 (insert, delete)

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	message	字符串	上传成功信息
400	err	字符串	错误信息

【GET】/data

用途：返回特定文件

参数格式：url parameter

变量名称	数据类型	描述
resource*	字符串	特型卡名称

filename*	字符串	文件名

返回值格式：multipart/form-data

返回码	变量名称	数据类型	描述
200	file	文件	对应文件
400	err	字符串	错误信息

【GET】/dataname

用途：取得对应类🄴已上传文件的名字

参数格式：url parameter

变量名称	数据类型	描述
resource*	字符串	特型卡名称

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	data	字符串列表	文件名列表
400	err	字符串	错误信息

【GET】/item

用途：取得特定 ID 的数据

参数格式：url parameter

变量名称	数据类型	描述
resource*	字符串	特型卡名称
key*	字符串	索引值

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	data	字符串	对应数据
400	err	字符串	错误信息

【GET】/key

用途：取得结构化数据的所有标签值。

参数格式：url parameter

变量名称	数据类型	描述
resource*	字符串	特型卡名称

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	data	字符串列表	标签列表
400	err	字符串	错误信息

4.4 搜索接口

搜索系统所使用的接口如下：

URL	用途
/search	搜索系统接口
/testes	倒排引擎微服务接口

【POST】/search
用途：给定句子查找符合条件的数据
参数格式：application/json

变量名称	数据类型	描述
query*	字符串	要搜索的句子

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	content	字符串列表	查找成功的数据
400	err	字符串	错误信息

【POST】/testes
用途：给定句子查找符合条件的数据
参数格式：application/json

变量名称	数据类型	描述
query*	字符串	要搜索的句子

返回值格式：application/json

返回码	变量名称	数据类型	描述
200	content	字符串列表	查找成功的数据

返回码	变量名称	数据类型	描述
400	err	字符串	错误信息

5 模块设计

5.1 搜索页面

搜索页面主要由以下模块构成：

package	用途
communication	通信接口模块
components	组件模块
views	页面模块
test*	测试模块

* 仅在单元测试时使用，对常规操作无影响。

5.1.1 communication

通信接口模块由以下函数组成，各部分的用途如下表所述：

函数名称	用途
query	调用后端的 POST /search 接口，查找句子

5.1.2 components

组件模块由以下部分组成，各部分的用途如下表所述：

组件	用途	子组件
BookCard	小说卡	ImageCarousel
CarCard	车系卡	CarNavTab,ImageCarousel
CarNavTab	车系导航栏	-
ImageCarousel	轮播图片	SearchBox
MedicineCard	医疗卡	MedicineNavTab,ImageCarousel
MedicineNavTab	医疗导航栏	-
PoemCard	诗词卡	ImageCarousel
SearchBox	搜索框	-

5.1.3 views

views 由以下页面构成，各部分在下表给出说明：

页面	用途	子组件
HomePage	主页	SearchBox
ResultPage	结果页	SearchBox,CarCard,BookCard, PoemCard,MedicineCard

5.1.4 test

测试由以下模块组成，各部分在下表给出说明：

文件 (.spec.js)	测试组件	测试项目
App	App	是否正确渲染
AxoisTest	Communication	路由返回不同值时系统的处理结果
BookCard	BookCard	传入数据后是否正确渲染页面
CarCard	CarCard	传入数据后是否正确渲染页面
CarNavTab	CarNavTab	传入不同数据后是否可以正确渲染组件
HomePage	HomePage	测试搜索框功能
ImageCarousel	ImageCarousel	传入不同数据后是否可以正确渲染组件
MedicineCard	MedicineCard	传入数据后是否正确渲染页面
MedicineNavTab	MedicineNavTab	传入不同数据后是否可以正确渲染组件
PoemCard	PoemCard	传入数据后是否正确渲染页面
SearchBox	SearchBox	测试搜索框功能与渲染结果

5.2 运营平台

运营平台主要由以下模块构成：

package	用途
communication	通信接口模块
components	组件模块
router	分配路由模块
store	当前用户信息存储模块
views	页面模块
test*	测试模块

* 仅在单元测试时使用，对常规操作无影响。

5.2.1 communication

通信接口模块由以下函数组成，各部分的用途如下表所述：

函数名称	用途
GET	处理朴素 GET 请求
GET_WITH_PARAMS	处理朴素含参 GET 请求
POST	处理 POST 请求
POST_Userlevel	处理 POST /login 请求，返回 UserLevel 数据
POST_User	处理 POST /login 请求，返回 User 信息
POST_File	处理 POST 请求，向后端发送数据

5.2.2 components

组件模块由以下部分组成，各部分的用途如下表所述：

组件	用途
PatternConfig/DeleteDialog	批量删除模板
PatternConfig/PostDialog	批量上传模板
PatternConfig/EditDialog	编辑现有模板
dynamicTags	动态标签，用于模板编辑与词表管理
resourceSelect	特型卡选择框

5.2.3 router

路由部分由以下组成，各部分用途如下表所述：

api	用途	组件
/home	主页	HomeView
/writesetting	写入行为描述	WScopy
/configpattern	模版配置	PatternConfig
/managedata	数据管理	DataManagement
/managedict	词表管理	DictManagement
/usermanage	用户管理	UserManagement
/userlogin	登录页	LoginPage

5.2.4 store

store 部分用来存储当前登录的用户的信息，方便其他组件调用：

模块	内容
admin	存储管理员信息
user	存储普通用户信息

5.2.5 views

views 由以下页面构成，各部分在下表给出说明：

组件	用途
DataManagement	数据管理
DictManagement	词表管理
HomeView	主页
LoginPage	登录页
NotFound	错误地址
PatternConfig	模板配置
UserManagement	用户管理
WScopy	写入行为描述

5.2.6 test

测试由以下模块组成，各部分在下表给出说明：

文件 (.spec.js)	测试组件	测试项目
App	App	是否正确渲染
AxoisTest	Communication	路由返回不同值时系统的处理结果
DeleteDialog	DeleteDialog	页面初始值，传入不同数据是否可以正确渲染
dynamicTags	dynamicTags	页面初始值，不同模板的渲染结果
EditDialog	EditDialog	页面初始值，传入不同数据是否可以正确渲染
placeHolder	placeHolder	页面初始渲染结果
PostDialog	PostDialog	页面初始值，传入不同数据是否可以正确渲染
resourceSelect	resourceSelect	页面初始值，传入不同数据是否可以正确渲染
ViewTest	(所有 views 页面)	页面组件渲染效果

5.3 后端

后端主要由以下模块构成：

package	用途
communication	通信接口模块
database	数据库模块
getter	数据整理模块
search	搜索模块
setter	数据调度/多线程模块
setup	数据处理模块
path*	文件路径模块
testcase*	测试数据模块

* 仅在单元测试时使用，对常规操作无影响。

5.3.1 communnication

通信接口模块由以下函数组成，各部分的用途如下表所述：

函数名称	用途
GetCategory	处理 GET /category 请求
GetConfig	处理 GET /setting 请求
GetDataName	处理 GET /dataname 请求
GetData	处理 GET /data 请求
GetItem	处理 GET /item 请求
GetKey	处理 GET /key 请求
GetTemplate	处理 GET /pattern 请求
AddUser	处理 POST /useradd 请求
AllUser	处理 POST /alluser 请求
DeleteUser	处理 POST /userdelete 请求
PostConfig	处理 POST /setting 请求
PostData	处理 POST /data 请求
PostEsTest	处理 POST /testes 请求
PostLogin	处理 POST /login 请求
PostSearch	处理 POST /search 请求
PostTemplate	处理 POST /pattern 请求
SetupRouter	路由初始化

其中，**SetupRouter** 函数的返回值是一个 gin 路由，即 *gin.Engine。其余函数的参数和返回值可见“通信接口设计”部分。

5.3.2 database

数据库模块由以下函数组成，各部分的用途如下表所述：

数据库	名称	函数名称	用途
Mysql	category	CreateCategoryTable	新建类名表格
		CountCategory	统计特型卡类🄵数量
		DropCategory	删除对应类🄵表格
		GetAllCategory	取出所有特型卡名
		InsertCategory	新增特型卡类🄵
	dict	CreateDatabase	新建数据库
		CreateTableInDict	创建表格
		DeleteByDocidFromDict	删除特定 docid 的所有数据
		DeleteTableFromDict	删除表格
		GetAllFieldFromDict	返回表中所有的字段名
		GetAllWordFromDict	返回表中所有数据
		InsertToDict	向表中插入数据
		SearchByDocidFromDict	返回指定 docid 的所有数据
		SearchByFieldFromDict	返回指定 field 的所有数据
		ShowTablesInDict	获取词典中所有表名
	docid	CreateDocidTable	新建表格
		DeleteDocid	删除指定 docid 对应的数据
		GetDocid	取出 docid 对应数据
		GetAllDocid	取出表格下所有数据
		InsertDocid	插入 docid 及对应数据
	data, config	CreateFileTable	新建文件表格
		DeleteFile	删除特定名字的文件
		GetFile	取出对应文字的数据
		GetFileName	取出表格下所有文件名
		InsertFile	插入文件
	UserInfo	AllUser	取出所有用户信息
		CreateTableForUserinfo	建立存储用户名和密码的表
		CreateTableForUserLevel	建立存储用户名和权限等级的表
		DeleteUser	根据用户名删除一个用户
		EncodePassword	密码再加密
		InsertPwdIntoSQL	向数据库中添加一个用户的信息
		SearchUser	根据用户名查找对应用户信息
		UserSignup	注册新用户
		UserSignIn	用户登录

数据库	名称	函数名称	用途
	pattern	CreateTableInPattern	新建表格
		DeleteFromPattern	删除表格
		DeleteTableFromPattern	删除对应模板
		FetchAllPattern	获取表中所有模板
		InsertToPattern	插入模板
ES		DeleteFromES	从指定的特型卡中删除数据
		FetchAllFromEs	从指定的特型卡中取出所有数据
		InsertToEs	向指定的特型卡中添加数据
		SearchFromEs	关键词搜索
		UpdateToEs	向指定的特型卡中更新数据
Redis		DeleteFromRedis	删除数据
		ExistInRedis	检查数据是否存在
		GetFromRedis	利用 docid 搜索数据
		SetToRedis	插入数据

5.3.3 getter

数据整理模块由以下函数组成，各部分的用途如下表所述：

函数名称	用途
GetConfig	返回特型卡对应的写入行为描述文件

5.3.4 path

文件路径模块由以下函数组成，各部分的用途如下表所述：

函数名称	用途
GetAbsPath	取得项目在容器内的绝对路径

5.3.5 search

搜索模块由以下函数组成，各部分的用途如下表所述：

服务	函数名称	用途
AC 自动机	BuildNewMatcher	指向给出的词典组成的 AC 自动机的指针
	Build	根据给出的词典建造自动机
	Check	检查句子是否成功匹配任意字词
	Match	取得句子成功匹配的字词
	NewMatcher	返回一个空的 AC 自动机指针
意图识别	BuildAC	建立 AC 自动机
	IntentionRecognition	识F搜索句子的意图
query 理解	Contains	查找字串是否在 slice 中出现过
	QueryUnderstanding	匹配模版并找出对应字词信息
倒排搜索, 摘要	Search	根据字词信息找出合适的数据

5.3.6 setter

数据调度/多线程模块由以下函数组成，各部分的用途如下表所述：

函数名称	用途
GetDocid	生成对应数据的 docid
SetConfig	根据写入行为描述文件，多线程全量建库
SetData	数据拆包，多线程处理
SetTemplate	存储模板配置文件
SetTemplateAll	在所有特型卡中存储模板配置文件
SetWord	存储用户配置的词典
SetWordAll	在所有特型卡中存储用户配置的词典

5.3.7 setup

数据处理模块由以下函数组成，各部分的用途如下表所述：

函数名称	用途
DeleteFileData	删除单一文件
DeleteItem	删除单一数据
DeleteResData	删除特定特型卡数据（全量建库用）

函数名称	用途
GetDocid	生成数据的唯一标识符
GetKey	取得单条数据的标签
SaveItem	取得数据所有子结点标签
StoreItem	根据写入行为将数据写入数据库中
UnpackXmlData	拆解 XML 形式的数据并入库
UnpackXmlFile	拆解 XML 形式的文件并入库
UpdateResData	更新特定特型卡数据（全量建库用）

5.3.8 testcase

测试数据模块由以下函数组成，各部分的用途如下表所述：

函数名称	用途
SetTestData	将单元测试需要的数据入库

6 数据库设计

6.1 简介

本次工程中，我们选择了 Mysql 数据库配合 mariadb 容器来储存常规数据。同时选择了 ElasticSearch 作为倒排引擎，方便我们反向索引数据的 docid。我们还使用了 Redis 存储每一个 docid 和对应的数据，加速搜索过程。

6.2 数据描述

6.2.1 Mysql

Mysql 数据库存储的数据有以下几类:

数据库名称	描述
category	特型卡类🄵
config	写入行为描述文件
data	文件形式上传之数据
dict	写入行为描述指定的词典
docid	储存单条结构化数据
pattern	存储用户配置的词典和模板
UserInfo	储存用户信息

相关的模型请参考模型描述。

6.2.2 ElasticSearch

对于每一条数据 (item)，会根据特型卡的不同而放入不同的 index 当中。然后将写入描述给出的内容以空格连接，再以 docid 为索引存入 ElasticSearch 当中。

6.2.3 Redis

Redis 的每条数据都有一个唯一索引值 (docid) 和对应内容。(处理好的摘要, JSON 格式)

6.3 模型描述

6.3.1 category

表名	槽位名称	数据类型	用途
word	category*	varchar(64)	所有特型卡的名字
(特型卡名)	category*	varchar(64)	指定特型卡数据的所有标签 (Tag)

6.3.2 file

file 模型用于写入描述 (config) 和数据 (data) 数据库中。

槽位名称	数据类型	用途
filename*	varchar(64)	文件名
data	mediumblob	数据内容

6.3.3 dict

dict 模型应用于由结构化文件提取的词以及用户维护的词典。

槽位名称	数据类型	用途
docid	varchar(100)	唯一标识符
field	varchar(100)	数据所属标签 (Tag)
word	varchar(100)	数据内容

6.3.4 docid

docid 模型以单条的形式储存所有原始结构化数据。

槽位名称	数据类型	用途
docid*	varchar(64)	唯一标识符
data	mediumblob	结构化数据内容
filename	varchar(64)	数据所属文件名

6.3.5 pattern

槽位名称	数据类型	用途
pattern*	varchar(150)	模板内容

6.3.6 UserInfo

表名	槽位名称	数据类型	用途
UserInfo	username*	varchar(100)	用户名
	userpassword	varchar(500)	用户密码（已加密）
UserLevel	username*	varchar(100)	用户名
	userlevel	varchar(10)	用户等级