目录

第一章 绪　论

1.应用开发背景及现状

搜索引擎是根据用户需求与一定算法，运用特定策略从互联网检索出制定信息反馈给用户的一门检索技术。搜索引擎依托于多种技术，如网络爬虫技术、检索排序技术、网页处理技术、大数据处理技术、自然语言处理技术等，为信息检索用户提供快速、高相关性的信息服务。搜索引擎技术的核心模块一般包括爬虫、索引、检索和排序等，同时可添加其他一系列辅助模块，以为用户创造更好的网络使用环境。截至 2020 年 3 月，我国搜索引擎用户规模达 7.50 亿，较 2018 年底增长 6883 万，占网民整体的 83.0%；手机搜索引擎用户规模达 7.45 亿，较 2018 年底增长 9140 万，占手机网民的 83.1%。

2. 应用开发目的及意义

随着搜索引擎系统的深入开发，越来越多的搜索引擎系统在给予用户诸多选择和便利的同时充斥着诸多低质量文章和劣质广告，致使用户产生诸多不适的体验感。针对于这一现象，本次 NoBugWiki 应用系统开发，我们致力于打造一款面向开发者解决开发问题的检索引擎系统。

3. 应用开发技术说明

3.1.Web 模块（前端技术）

3.1.1.HTML

超文本标记语言（英语：HyperText Markup Language，简称：HTML）是一种用于创建网页的标准标记语言。

3.1.2.CSS

层叠样式表（英语：Cascading Style Sheets，缩写：CSS；又称串样式列表、级联样式表、串接样式表、阶层式样式表）是一种用来为结构化文档（如 HTML 文档或 XML 应用）添加样式（字体、间距和颜色等）的计算机语言，由 W3C 定义和维护。

3.1.3.JavaScript

JavaScript（通常缩写为 JS）是一种高级的、解释型的编程语言。JavaScript 是一门基于原型、头等函数的语言，是一门多范式的语言，它支持面向对象程序设计，指令式编程，以及函数式编程。

3.1.4.Vue.js

Vue.js（/vju:/，或简称为 Vue）是一个用于创建用户界面的开源 JavaScript 框架，也是一个创建单页应用的 Web 应用框架。2016 年一项针对 JavaScript 的调查表明，Vue 有着 89%的开发者满意度。在 GitHub 上，该项目平均每天能收获 95 颗星，为 Github 有史以来星标数第 3 多的项目。

3.2.Web 模块（后端技术）

3.2.1.Java

Java 是一种广泛使用的计算机编程语言，拥有跨平台、面向对象、泛型编程的特性，广泛应用于企业级 Web 应用开发和移动应用开发。

3.2.2.SpringBoot

Spring Boot 是在 Spring 框架基础上创建的全新框架。 相比于以往的一些开发框架，Spring Boot 使用更加简单，而且功能更加丰富，性能更加稳定而健壮。

3.2.3.Mybatis-Plus

Mybatis-Plus 是一个基于 Java 的持久层框架。iBATIS 提供的持久层框架包括 SQL Maps 和 Data Access Objects（DAOs）

3.2.4.Maven

Maven 项目对象模型(POM)，可以通过一小段描述信息来管理项目的构建，报告和文档的项目管理工具软件。

3.3.消息队列模块（Kafka 技术）

Kafka 是由 Apache 软件基金会开发的一个开源流处理平台，由 Scala 和 Java 编写。该项目的目标是为处理实时数据提供一个统一、高吞吐、低延迟的平台。其持久化层本质上是一个“按照分布式事务日志架构的大规模发布/订阅消息队列”，[3]这使它作为企业级基础设施来处理流式数据非常有价值。此外，Kafka 可以通过 Kafka Connect 连接到外部系统（用于数据输入/输出），并提供了 Kafka Streams——一个 Java 流式处理库。

3.1.5.Element-Plus

Element Plus，一套为开发者、设计师和产品经理准备的基于 Vue 3.0 的桌面端组件库。

3.1.6.Node.js

Node.js 是能够在服务器端运行 JavaScript 的开放源代码、跨平台运行环境。Node.js 由 OpenJS Foundation （原为 Node.js Foundation，已与 JS Foundation 合并）持有和维护，亦为 Linux 基金会的项目。

3.1.7.Electron

Electron（原名为 Atom Shell[5]）是 GitHub 开发的一个开源框架。它通过使用 Node.js（作为后端）和 Chromium 的渲染引擎（作为前端）完成跨平台的桌面 GUI 应用程序的开发。Electron 现已被多个开源 Web 应用程序用于前端与后端的开发，著名项目包括 GitHub 的 Atom 和微软的 Visual Studio Code。

3.4.数据采集模块（ Python、Scrapy-Redis）

Python 是一种广泛使用的解释型、高级和通用的编程语言。Python 支持多种编程范型，包括函数式、指令式、结构化、面向对象和反射式编程。它拥有动态类型系统和垃圾回收功能，能够自动管理内存使用，并且其本身拥有一个巨大而广泛的标准库。
Scrapy 是一个用 Python 编写的自由且开源的网络爬虫框架。它在设计上的初衷是用于爬取网络数据，但也可用作使用 API 来提取数据，或作为生成目的的网络爬虫。该框架目前由网络抓取的开发与服务公司 Scrapinghub 公司维护。

3.5.实时流数据计算模块（Flink、 Python）

Apache Flink 是由 Apache 软件基金会开发的开源流处理框架，其核心是用 Java 和 Scala 编写的分布式流数据流引擎。Flink 以数据并行和管道方式执行任意流数据程序，Flink 的流水线运行时系统可以执行批处理和流处理程序。此外，Flink 的运行时本身也支持迭代算法的执行。

Flink 提供高吞吐量、低延迟的流数据引擎以及对事件-时间处理和状态管理的支持。Flink 应用程序在发生机器故障时具有容错能力，并且支持 exactly-once 语义。程序可以用 Java、Scala、Python 和 SQL 等语言编写，并自动编译和优化到在集群或云环境中运行的数据流程序。

3.6.数据存储模块（Mysql、 Redis）

MySQL 在过去由于性能高、成本低、可靠性好，已经成为最流行的开源数据库，因此被广泛地应用在 Internet 上的中小型网站中。随着 MySQL 的不断成熟，它也逐渐用于更多大规模网站和应用，比如维基百科、Google 和 Facebook 等网站。非常流行的开源软件组合 LAMP 中的“M”指的就是 MySQL。
Redis 是一个使用 ANSI C 编写的开源、支持网络、基于内存、分布式、可选持久性的键值对存储数据库。从 2015 年 6 月开始，Redis 的开发由 Redis Labs 赞助，而 2013 年 5 月至 2015 年 6 月期间，其开发由 Pivotal 赞助。在 2013 年 5 月之前，其开发由 VMware 赞助。根据月度排行网站 DB-Engines.com 的数据，Redis 是最流行的键值对存储数据库。

技术上使用 Redis 的优点：

一．Redis 性能极高，能读的速度是 110000 次/s,写的速度是 81000 次/s ；

二．丰富的数据类型 – Redis 支持二进制案例的 Strings, Lists, Hashes, Sets 及 Ordered Sets 数据类型操作；

三．原子 – Redis 的所有操作都是原子性的，同时 Redis 还支持对几个操作全并后的原子性执行；

四．丰富的特性 – Redis 还支持 publish/subscribe, 通知, key 过期等等特性；

五．Redis 支持数据的持久化，可以将内存中的数据保持在磁盘中，重启的时候可以再次加载进行使用；

六．Redis 不仅仅支持简单的 key-value 类型的数据，同时还提供 list，set，zset，hash 等数据结构的存储。

第二章 功能需求说明

1. 用户需求分析

1.1. 价值需求分析

一、通过搜索内容的实时筛选，高效信息计算，输出有用的信息，并根据用户所提供的关键字，实时推送相关文献可供用户选择，帮助用户学习；
二、根据已注册用户使用数据情况的可视化，通过个性化热门文章推荐，贴心地为用户推送所需的相关文献，使用户在提升专业知识的同时感受大数据的魅力；
三、在学习/工作事务管理模块上运用简单快捷的 Todo 管理，标签式内容，保留用户所想保留的内容，便于用户的查找和使用，予以用户省时省力省心的体验感；
四、认真对待用户的每一次查询，以强大的技术服务保证提供高效的信息检索服务，助力开发者成长之路；
五、深度兼容国产操作系统 Deepin，给予用户熟悉感的同时带来一点个性创意。

1.2. 数据需求分析

该引擎系统通过对传统搜索引擎和各大开发社区进行实时数据采集，并对其数据进行二次聚合与与筛选、精排；此外在搜索过程、搜索结果、探寻式搜索表现其 3 个方面对每个用户进行全方位观察，根据用户行为信息，建立用户模型，通过实时流计算实现实时用户内容精准推荐；此外通过数据可视化实现用户自览，通过 todo 事项与内容标签化实现内容高度灵活性。

2. 功能需求分析

2.1. 系统功能分析

2.1.1. web 系统

图 2.1

如图 1-1 该 web 系统主要通过页面的检索，发送给后台服务器，通过消息缓存集群，缓存数据，在不妨碍用户操作的同时，通过调用数据缓存，加快数据搜索速度；在自动筛选可用的信息给予用户多种选择的同时提高用户的学习效率，并且加深用户对其相关知识的深入了解和学习；通过查询已注册用户的在线使用状态，对用户行为进行可视化操作的同时，及时推送用户所需相关联的技术知识。

2.1.2. 数据采集

在互联网行业快速发展的今天，数据采集已经被广泛应用于互联网及分布式领域，数据采集领域已经发生了重要的变化。数据采集，又称数据获取，是利用一种装置，从系统外部采集数据并输入到系统内部的一个接口。本次引擎系统的数据采集运用开发语言 Python 和数据采集框架 Scrapy-Redis，通过用户的实时采集数据，依赖于数据采集集群，从图 1-2 中各个网站采集的资源实行热门内容的更新

2.1.3. 数据流计算

本次系统数据流采用的是需求驱动法。通过检索行为、点击行为、预览停留时间等需求驱动时，开始数据流计算，充分发挥了数据流的并行性，简化了代码，节约了缓存

2.1.4. 消息队列

消息队列（Message Queue）是一种应用间的通信方式，消息发送后可以立即返回，有消息系统来确保信息的可靠专递，消息发布者只管把消息发布到 MQ 中而不管谁来取，消息使用者只管从 MQ 中取消息而不管谁发布的。本次项目消息队列的实现通过打卡情况任务信息和数据采集任务信息，组成消息队列，依赖于消息队列集群，把消息反馈到服务器，从而给予用户响应

2.1.5. 数据持久化

数据持久化顾名思义就是把程序中的数据以某种形式保存到某存贮介质中，以达到持久化的目的。本次系统针对于数据的持久化采用的方式是依赖于数据库，从而达到数据的持久化保存

3. 可行性分析

基于微服务、消息转发、分布式爬虫、流计算、云存储五种架构可提供稳定服务保障；

第三章 概要设计说明

1. 系统模块图

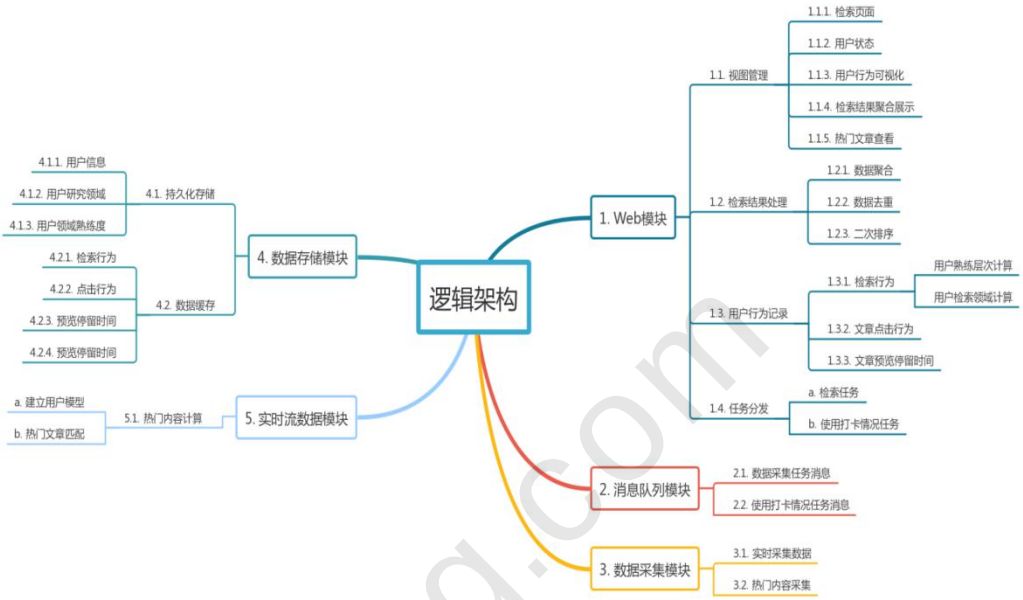

图 3.1 系统模块图

2. 功能需求逻辑图

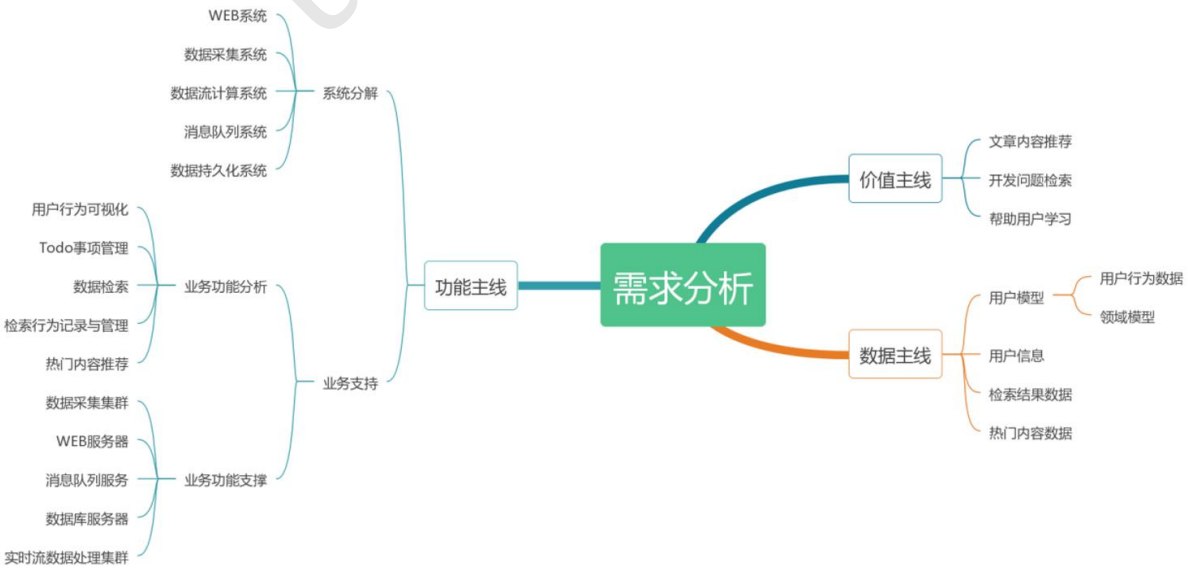

图 3.2 功能需求逻辑图

3.数据存储分析

3.1. 数据化结构储存

数据化的存储结构该引擎系统项目运用数据化持久的 Mysql 技术（MySQL 在过去由于性能高、成本低、可靠性好，已经成为最流行的开源数据库），因整体数据存储需求很小，对于结构化数据来说内容更少，所以在该项目中结构化数据只有用户消息表（图 3.4）与用户领域检索次数（图 3.3）两个表。

用户领域检索次数表 direction				
字段名称	字段含义	数据类型	NULL	默认值
Email	邮箱	varchar(15)	No null	0000000000
Backend	后端开发	int(10)		0000000000
Linuxom	Linux 运维	int(10)		0000000000
Cloudbigdata	云计算与大数据	int(10)		0000000000
database_d	数据库	int(10)		0000000000
netsecurity	系统安全	int(10)		0000000000
frontend	Web 前端	int(10)		0000000000
computermajor	计算机专业	int(10)		0000000000
ai	人工智能	int(10)		0000000000

图 3.3

用户消息表 user				
字段名称	字段含义	数据类型	NULL	默认值
Email	用户邮箱	varchar(20)	No null	
Name	用户名称	varchar(20)	No null	
Password	用户密码	varchar(20)	No null	

图 3.4

3.2. 非数据化结构储存

数据化的存储结构该引擎系统项目运用数据缓存 Redis（ Redis 是一个使用 ANSI C 编写的开源、支持网络、基于内存、分布式、可选持久性的键值对存储数据库）。

对于非结构化数据，该项目采用键值对（”key = value”）形式存储，针对性强，扩展性高。

如图 2-5（非数据化存储图），当记录用户行为类型缓存时，用户点击数据，key 为用户注册时的邮箱信息加上点击的数据信息，value 返回的是该时间轴上用户的行为信息，存入缓存流；当用户预览文档停留时，key 为用户邮箱信息和停留时间，value 为该 key 对应时间轴的行为信息。以此类推，把用户行为类型缓存分为四种情况：用户点击数据行为、用户预览停留时间行为、用户搜索数据行为、用户检索日志行为。把事务管理缓存分为两种情况：一种是检索时爬虫到的数据随机生成的工作 ID，产生的 value 为用户检索词，存入缓存待后续使用时直接调用；一种是爬虫到的结果，根据之前爬虫到缓存中的检索词使得 key 为用户检索词，value 为检索结果，方便后续反馈给用户查看。

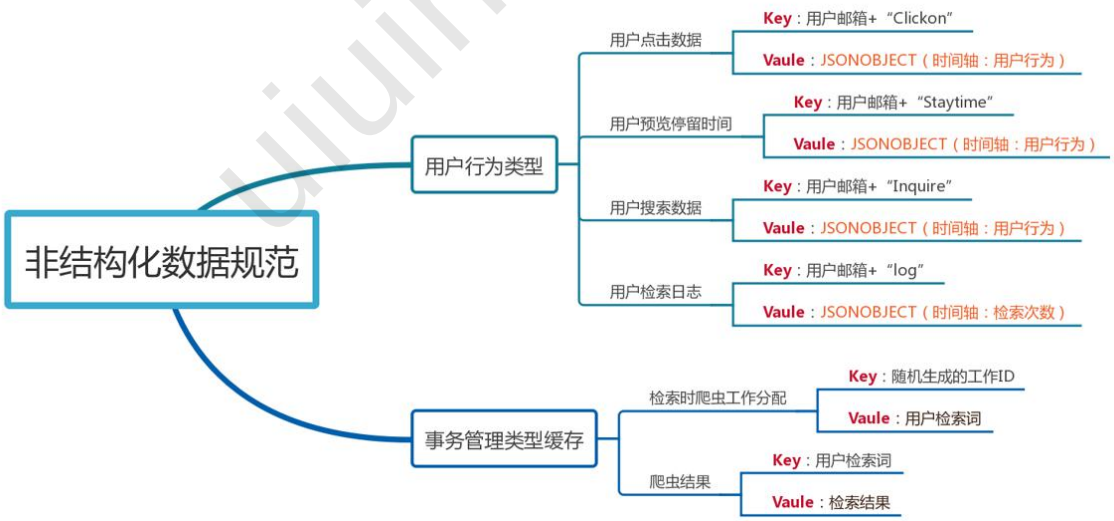

图 3.5 非数据化存储图

第四章 详细设计说明

1.web 项目模块

1.1 检索页面

通过图 4.1 可知，用户通过搜索关键字，使得 web 服务器对其进行数据处理。为了防止服务器资源故障，通过消息队列集群，分配任务到数据采集集群，从图 4.2 中采集数据源，并从中通过比较、筛选出有用信息反馈给消息队列集群，集群再回馈信息给 web 服务器；同时通过使用分布式存储器，用户在第一次使用时，通过简要分析将数据存储到缓存中，当再次搜索时该行为记录被检测到则可以直接从缓存流中调用，不需要再次通过集群搜索，大大地提高了搜索速度，降低了内存空间的消耗。

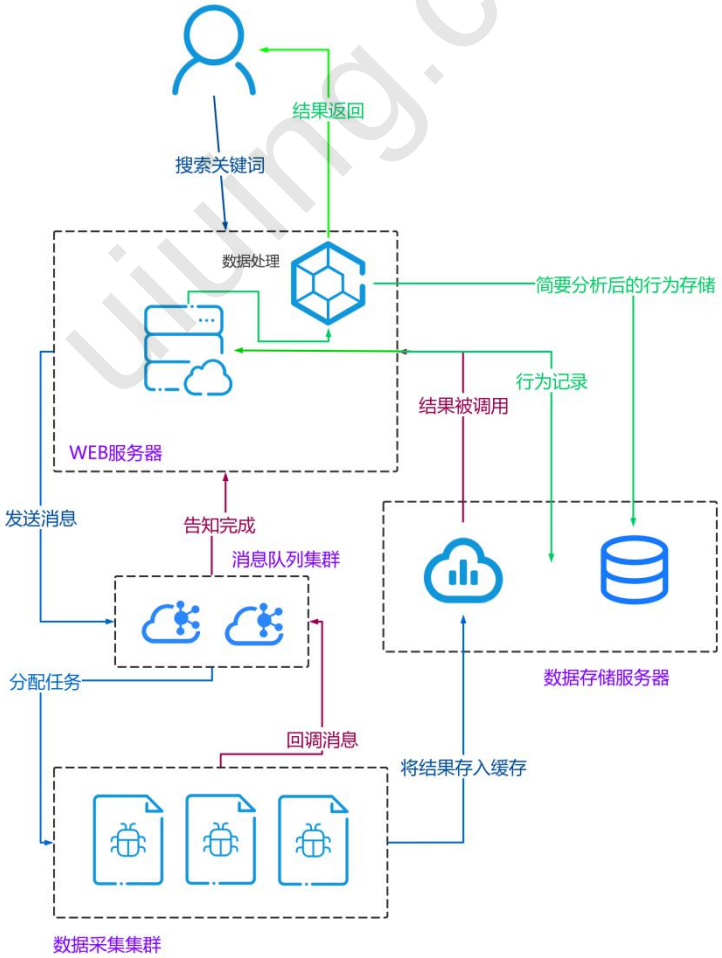

图 4.1 检索页面流程图

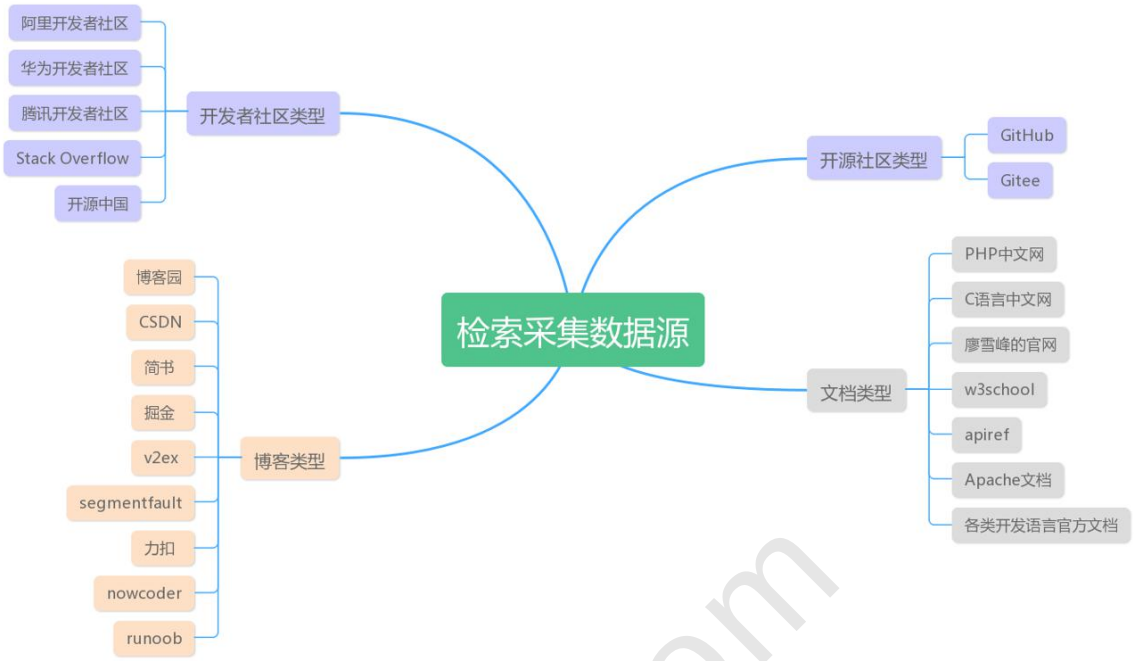

图 4.2 采集数据流程图

以博客 、开发者社区、开源社区、文档四种类型为数据源方向，从四种类型的网站中采用网络爬虫技术、检索排序技术、大数据处理等技术进行数据的收集。

1.2.用户行为下的可视化

据图 4-3 可知，web 服务器通过监控用户的行为，通过分布式储存服务器储存到缓存流中去，建立用户模型，基于知识结构对其推算，预演，在调用用户行为数据时通过实时流计算集群，反馈给用户并存入用户推荐内容，通过之前在数据采集集群中收集的数据定时更新推荐信息，等待用户再次搜索时，方便用户直接调用。

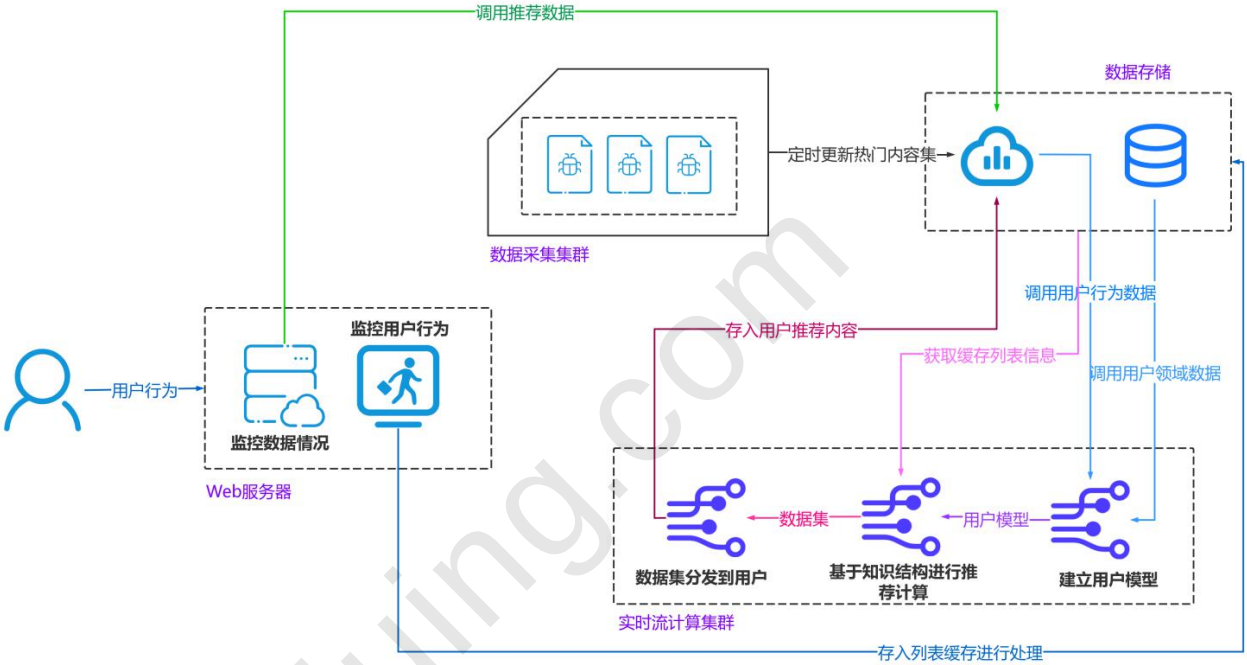

图 4.3 用户行为可视化流程图

1.3.web 层框架结构图

Web 层框架的实现由展示层、接口网关、业务层、应用层、和存储结构组成。各个层次基于 SpringBoot 的部署，用户通过登录已注册的账号进入系统，由 VUE.js 和 Elment Plus 等前端技术获取页面信息，由 Post 或 Get 发送请求或者由 Nginx 代理来得到用户的请求，业务层通过分布式存储器根据用户行为调用相关信息，通过数据采集集群分工完成数据的筛选和热门文章内容推送的更新，由应用层回馈反应，这四层结构都有 Shiro 权限控制阻止未注册或未登录用户的使用。

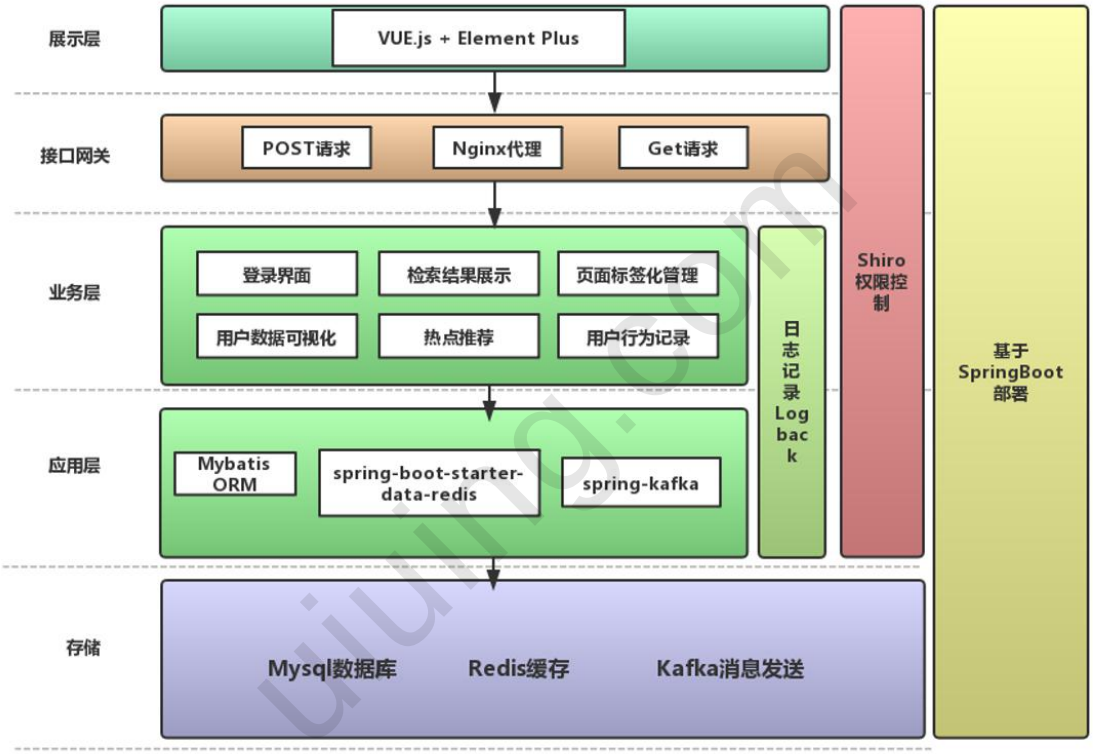

图 4.4web 层框架结构图

2. 消息队列模块

如图 4.5 消息队列流程图本次消息队列模块依赖分布式消息队列，以集群的方式，通过实行打卡情况任务消息和数据采集任务消息的收集组成消息队列，把消息反馈到服务器，从而给予用户响应。

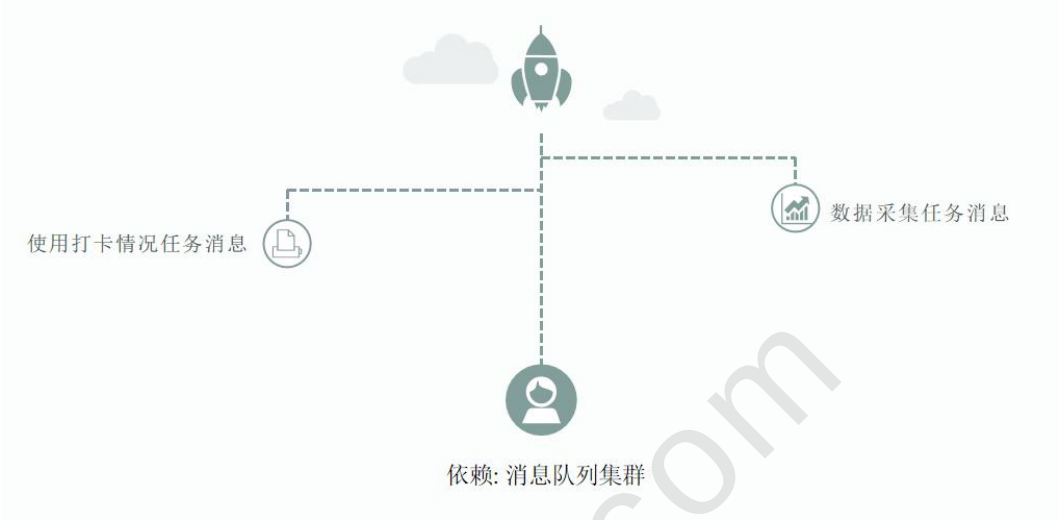

图 4.5 消息队列流程图

3. 数据采集模块

如图 4.6 本次数据采集模块，通过进程管理程序进行工作分配，实现任务对接。在任务对接过程中，实现定时数据和实时数据的收集以及消息的接收和回馈等业务。同时，该项目采用 Scrapy-redis 技术(Scrapy-Redisjis 技术是一个基于 Redis 的 Scrapy 分布式组件。它利用 Redis 对用于爬取的请求(Requests)进行存储和调度(Schedule)，并对爬取产生的项目(items)存储以供后续处理使用。)在一定程度上，scrapy-redis 技术重写了 scrapy 一些比较关键的代码，将 scrapy 变成一个可以在多个主机上同时运行的分布式爬虫，在数据的爬取上更快，搜索时间更短，消耗的内存更低。在解决运行环境和配置问题上，该项目采用 Docker 容器 ，该容器将软件打包成标准化单元，以用于开发、交付和部署，同时 Docker 容器赋予了系统项目独立性，使其免受外在环境差异的影响，从而方便于做持续集中并有助于整体发布的容器虚拟化技术。

任务对接	工作分配/进程管理程序		
业务	定时数据收集	实时数据收集	消息接收与回馈
技术实现	Scrapy-redis		
虚拟化	Dorker 容器		
基础服务	数据采集集群		

图 4.6 数据采集表格详列

4. 实时流数据计算模块

该模块运用分布式处理引擎：Flink、开发语言：Python。分布式处理引擎 Flink 采集数据源，Python 语言用于数据的爬取，两者技术的结合，对缓存中的数据进行数据的采集、传输、存储及展示，便于用户再次搜索时该行为记录被检测，则可直接从缓存流中调用，不需要再次通过集群搜索，大大地提高了搜索速度，降低了内存空间的消耗。

5.数据存储模块

如图 4.7，为保证数据的稳定性和空间的充分利用性，本次数据存储模块采用数据化存储结构和非数据化存储结构两种形式存储数据，数据化存储结构采用 mysql 技术，非数据化存储结构采用 Redis 技术。

图 4.7 数据存储模块图

第五章 界面设计说明

1. 信息检索

输入内容页面

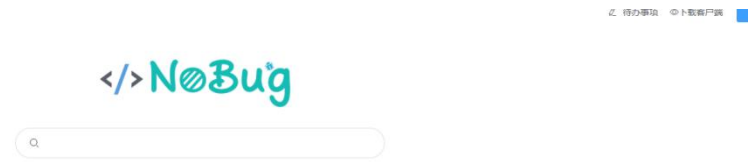

图 5.1

检索结果菜单页面

图 5.2

检索结果预览案例

图 5.3

2. 个性化热门推荐

单击菜单栏，出现【热点】选项卡出现在此内

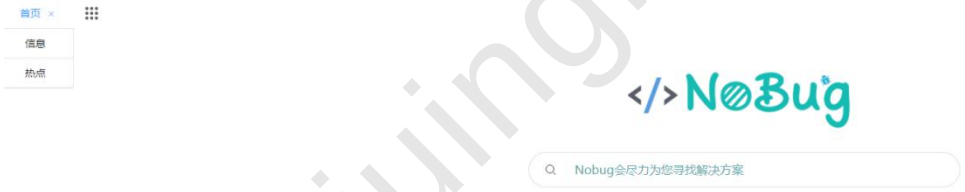

图 5.4

与检索结果页面一致，此为目录

图 5.5

内容预览

图 5.6

3. 内容管理

3.1 登陆注册

图 5.7 登录

图 5.8 注册

3.2 用户信息管理

用户可自定义程序线程程度，优化程序速度，此为页面

图 5.9

用户自定义后台播放音乐音源

图 5.10

3.3 标签化管理内容

每次检索都会记录在菜单栏中，存为标签

图 5.11

点击右边【X】即可关闭该页面

细化到具体内容也是如此

图 5.12

4. 开发/使用统计分析

4.1 热力图

图 5.13

4.2 饼图

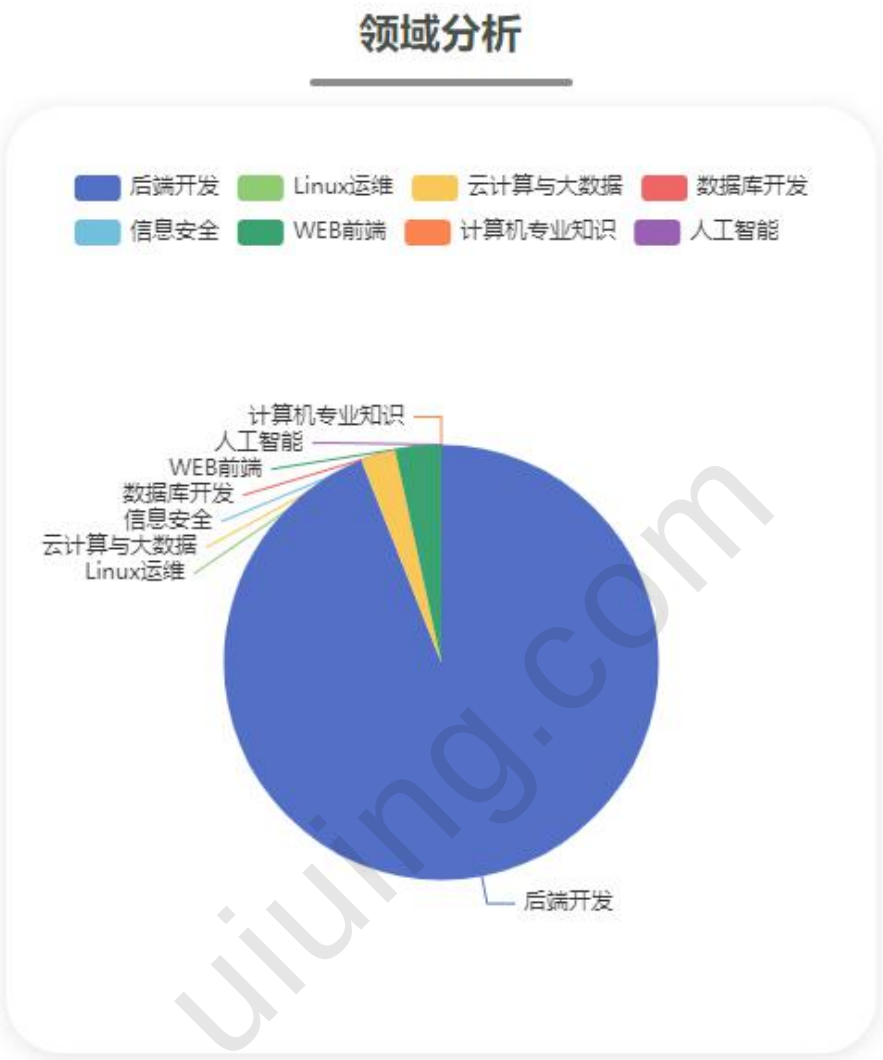
领域分析
后端开发
Linux运维
云计算与大数据
数据库开发
信息安全
WEB前端
计算机专业知识
人工智能
计算机专业知识
人工智能
WEB前端
数据库开发
信息安全
云计算与大数据
Linux运维
后端开发

图 5.14

4.3 词云

图 5.15

4.4 待办事项

如图 5.16，待办事项页面进入按钮在此处

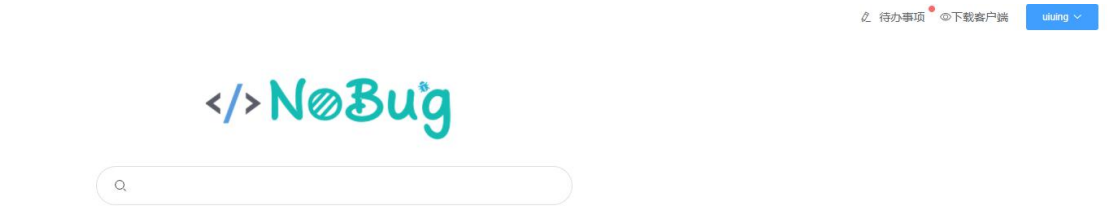

图 5.16

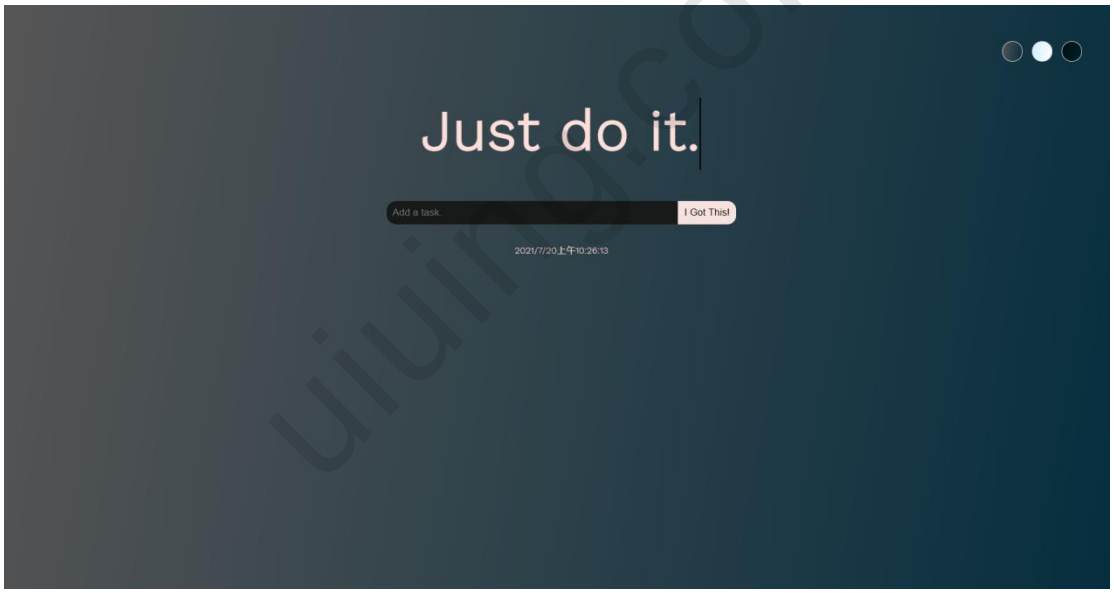

图 5.17 待办事项首页

5. 下载首页

图 5.18

6. 页面结构

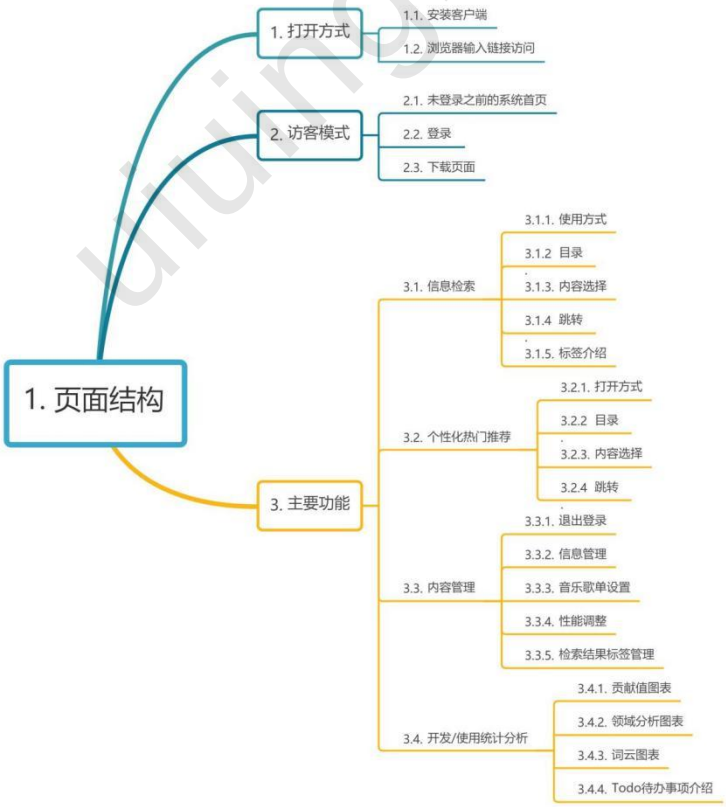

图 5.19 页面结构思维导图

第六章 用户操作手册

1. 使用环境

1.1 终端设备

适配所有具备图形化界面的终端

1.2 操作系统

WEB 访问：适配所有操作系统（包括移动设备、电脑设备）
客户端使用：

Windows7 及以上版本（兼容 x86、64）
Mac os 10.13 及以上版本
基于 Debain/Arch 发行版本的 Linux 操作系统

1.3 WEB 访问浏览器内核说明

兼容所有浏览器内核，出于体验考虑限制 ie9 浏览器访问
经测试，以下浏览器可完美运行：

Chrome 浏览器、Firefox 浏览器、IE11 浏览器、Safari 浏览器、Opera 浏览器、360 浏览器、猎豹浏览器、搜狗、遨游、QQ 浏览器、百度浏览器、2345 浏览器

1.4 网络环境

请在最低宽带 1M 环境下访问

2. 使用注意

2.1 环境漏洞

受发行限制，固安装客户端将可能受到操作系统以及防火墙软件限制，如果遇到请尽量避免，说明该安装程序为安全程序
该漏洞不影响客户端安装后使用体验

2.2 意外漏洞

为积极保障开发与扩展，该项目服务通信采用 http 协议，而非 https 协议，固浏览器访问将可能出现安全提醒，请尽量忽略，不允许数据安全与使用体验

2.3 服务漏洞

如果出现内容加载缓慢，可能因为访问人数过过，服务器过载导致，请稍等片刻，让服务器缓冲一下；如果问题长久存在，请检查是否为电脑原因，为适应用户需求，使用者可前往项目的【用户信息管理中心】调整内容加载其线程程度，如图 6.1

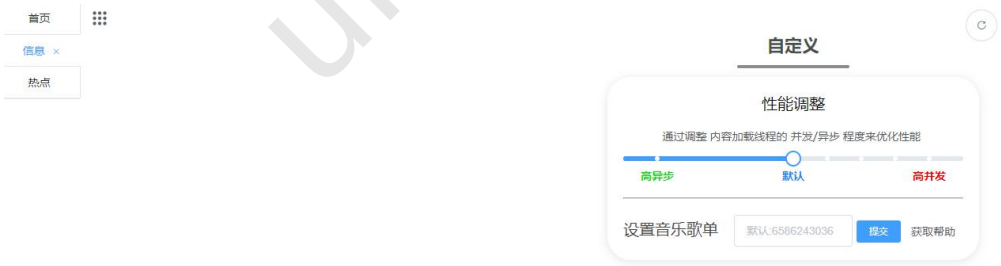

图 6.1

3. 操作事项

3.1 关于检索

用户检索使用时如果出现结果返回过慢或者 404 情况，原因可能出在服务器过载或者实时采集任务正在排队，请耐心等待
当然这种情况很少，但要注意的是尽量想好再检索，在我们内容聚合与缓存机制下，这将给您带来更好的体验

3.2 关于菜单栏

菜单栏内除基础三项服务（首页、信息、热点）以外，所有标签可自由管理，如果发生问题也不存在连锁反应，所有不会影响您的继续使用

3.3 待办事项

待办事项为独立页面，再客户端下使用情况最佳
注：用户退出登录以后将清除待办事项内容

3.4 用户信息

用户信息所有图表均可鼠标点击查看其具体内容

4. 隐私协议

4.1 用户信息

该项目的一切用户凭证仅依赖注册邮箱与密码，不存在任何网络信息的获取

4.2 后台分析

后台仅根据用户的检索的行为进行分析

第七章 安装手册

1.WEB 方式访问

打开浏览器，输入: nobug.wiki 即可访问

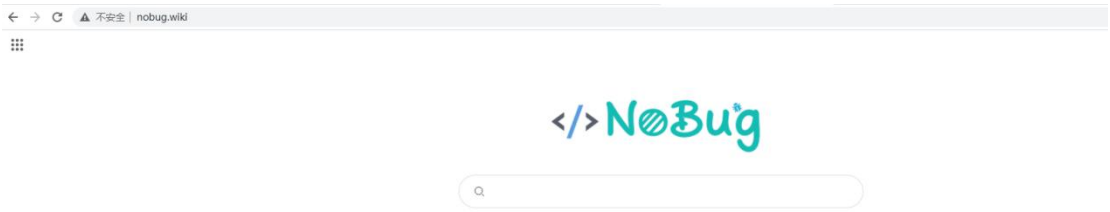

图 7.1 输入示范

2.安装客户端

2.1 下载客户端

打开浏览器，输入: nobug.wiki 访问主页，请参考图 7.1

之后点击右上方的【下载客户端】选项，进入下载页面

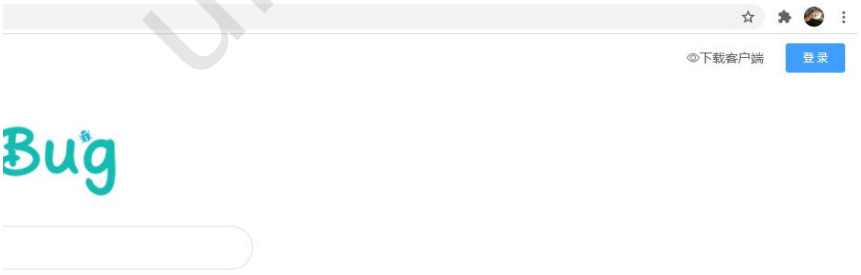

图 7.2 下载客户端位置

图 7.3 下载页面

进入下载页面之后，点击对应的操作系统按钮进行下载，下载情况如图 7.4

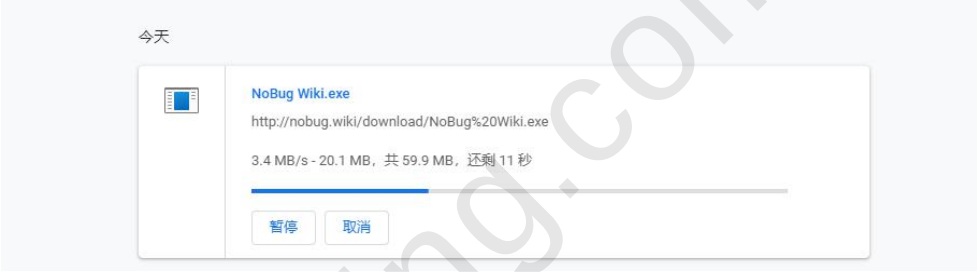

图 7.4 下载中样例

2.2 安装客户端

以 Windows 系统为案例，使用者只需单机下载完成的文件即可进入安装引导界面

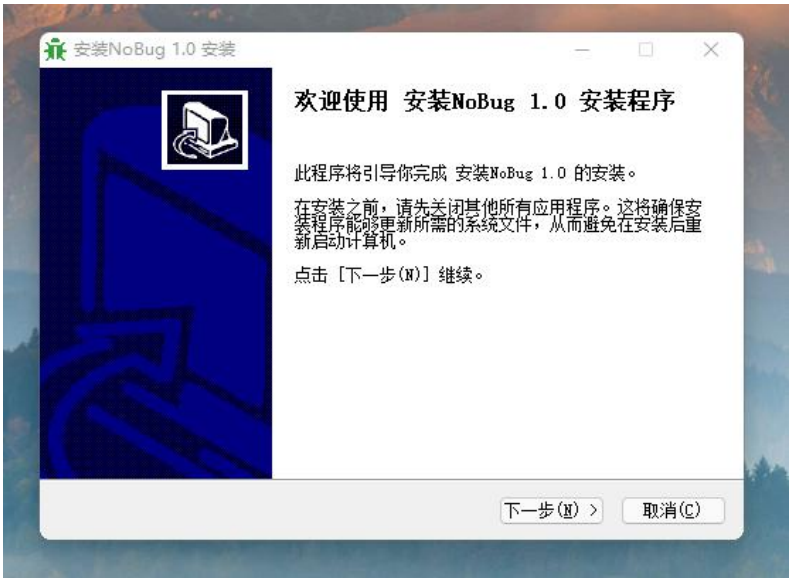

图 7.5 安装引导界面

点击下一步，同意用户协议

图 7.5 用户协议许可界面

之后按引导一直点击下一步即可

图 7.6 安装结束页面

在此之后，双击如图 7.7 所示桌面快捷方式，或者图 7.8 所示菜单地址，即可打开

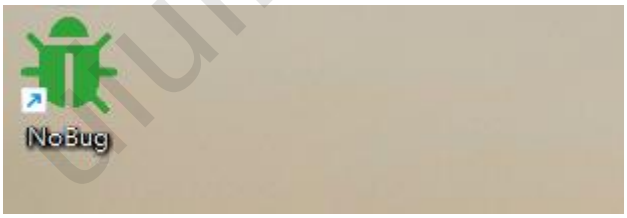

图 7.7 桌面生成快捷方式

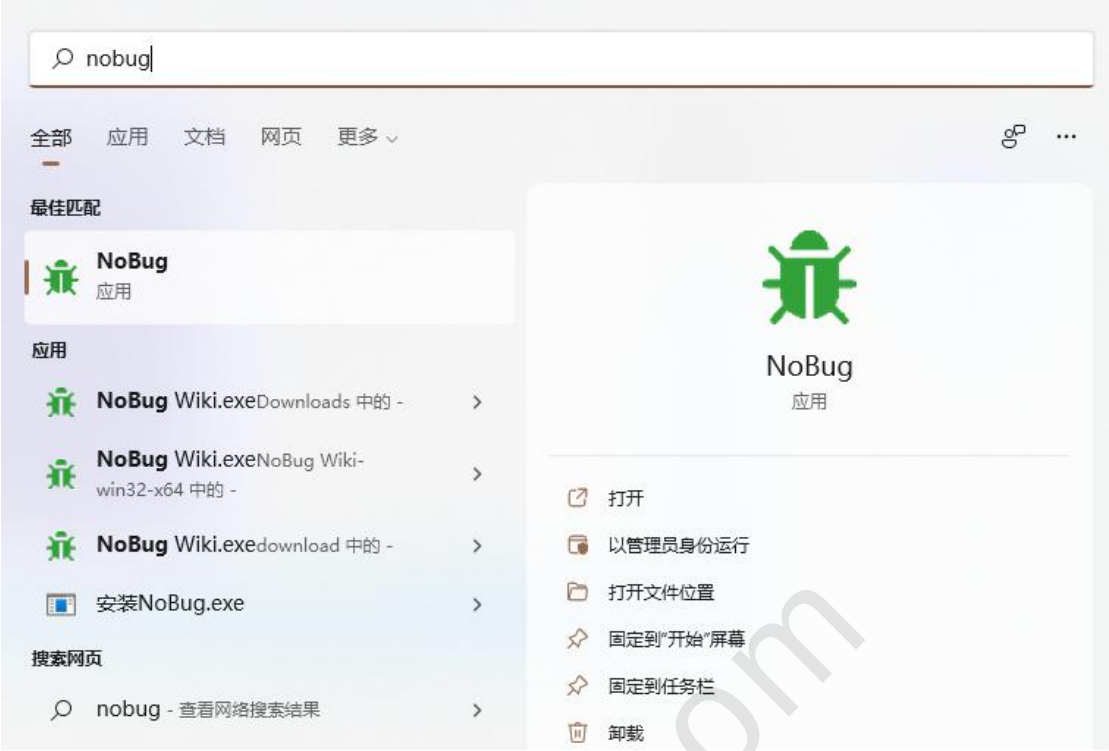

图 7.8 菜单栏检索应用

第八章 总结

1. 作品简介

NoBugWiki 是一个面向个人开发者的开发助理工具，具备一套完整的处理开发者的开发问题的解决方案

提供：信息检索、个性化热门推荐、开发/使用统计分析、内容管理，四种功能服务；

基于：微服务、消息转发、分布式爬虫、流计算、云存储五种架构提供稳定服务保障；

通过：用户行为分析、实时内容采集、开发知识模型、内容二次分析保障用户需求能更好得到解决方案

2. 作品功能及效果展示

作品四大功能：信息检索、个性化热门推荐、开发/使用统计分析、内容管理
信息检索：用户登录后可到项目首页搜索框内输入开发问题，之后将跳转至查询结果页面
个性热门推荐：用户可以查看经过个性化计算的热门内容推荐
开发/统计/使用统计分析：用户可以看到自己使用的情况，并且会根据后台的算法制作出其使用分析图标
内容管理：用户可对自己账户进行管理，并且内容包括：账户注销、todo 待办事项、检索结果标签化管理

3. 作品安装说明

作品提供两种使用方式
第一种为 WEB 页面访问：用户访问链接：http://nobug.wiki 即可直接访问；
第二种通过安装使用：像市面上的大部分软件一样，开箱即用。
在此我们提供了三种客户端的安装包，分别是：Windows、Linux、Mac os，分别细化有：Windows7
及以上版本、mac os 10.13 及以上、基于 Debain/Arch 发行版的 Linux 系统，
安装包下载方式：
1、访问链接：http://nobug.wiki/download/index.html 下载

2、提交文件中可安装软件压缩包中有安装包，选择对应系统即可

4. 设计思路

该作品基于五种架构服务：微服务、消息转发、分布式爬虫、流计算、云存储。
微服务：提供 web 服务，包括上面的作品功能及效果展示在内，还包括：用户管理、用户检索任务分配、用户存活监控，用户行为统计，检索结果二次处理
消息转发：负责微服务和分布式爬虫两架构之间的通信
分布式爬虫：负责实时抓取用户检索词内容、定时热门内容采集、任务接收
流计算：负责建立用户模型、计算出用户个性化热门推荐内容
云存储：数据持久化服务器负责存储用户信息，数据缓存服务器负责存储一切需要存储数据

5. 设计重难点

重点：
1、实时内容筛选，高效信息计算，只输出有用的信息；
2、个性化热门文章推荐，解决问题的同时，能不断提高认知；
3、学习/工作事务管理，Todo 事项管理，标签式内容，只保留想保留的；
4、根据用户行为进行数据可视化，从每一次行为中挖掘用户闪光点；
5、认真处理每一次查询，从用户角度出发，以强大的技术服务作保证每一次都提供高效的信息检索服务，助力开发者成长之路；
6、兼容各终端；
7、高度可扩展性，性能可扩展

难点：
1、负责实时内容爬虫模块的性能优；
2、个性化内容推荐的数据源与处理依据；
3、网站内容如何标签化，如何提高定制化；
4、如何保障程序实用性；
5、如何保证代码生命周期