聚合搜索

项目中使用了哪些技术栈，分别介绍Spring Boot Elastic Stack的作用

项目技术栈:SSM+Spring Boot、iElastic Stack、MySQL、MyBatis-Plus、Jsoup、Hutool工具库。
Spring Boot:用于快速构建基础的后端项目，只需要修改配置文件，就能轻松整合 SSM、MySQL、Elasticsearch 等依赖。
Elastic Stack:包括 Elasticsearch 搜索引擎、Kibana 可视化看板、Logstash 数据传输
。Elasticsearch:存储文章数据，提高根据关键词搜索文章内容的灵活性
Kibana:可视化 Elasticsearch 存储的数据，并能通过 Dev Tools 对 Elasticsearch 进行操作调试
。Logstash:将文章数据定时从 MySQL同步到 Elasticsearch

二次开发了Spring Boot初始化模版，这个模版有哪些功能

主要功能如下
整合常用的组件依赖，比如MySQL、Redis、Elasticsearch、Hutool等。Spring Session Redis 分布式登录
全局请求响应拦截器
全局异常处理器
自定义错误码
·封装通用响应类
·Swagger + Knife4j接口文档
自定义权限注解 + 全局校验.
全局跨域处理
长整数丢失精度解决
用户相关业务:用户登录、注册、注销、更新、检索、权限管理·帖子相关业务:帖子创建、删除、编辑、更新、数据库检索、ES 灵活检索
。文件上传

介绍HttpClient，如何使用其来抓取外部网站的文章，简述整个过程?

Apache HttpClient 是用于发送 HTTP 请求并处理 HTTP 响应的工具库，类似的还有 Hutool的HttpRequest、OKHttp等
首先我通过 F12 控制台捕获到了目标网站加载文章数据的 HTTP 请求，然后使用 HttpClient在 Java 代码中构造请求，比如指定请求类型、请求头、携带查询文章的请求参数等。发送请求后，判断响应状态码，如果响应正常，则将响应的JSON 字符串转换为响应实体类，对数据进行一定的处理后写入到业务数据库中。

什么是Jsoup，它和HttpClient有什么区别?

二者是截然不同的两个 Java 库!Jsoup 是一个解析和操作 HTML 文档的工具库，可用于网络爬虫，能够从网页上抓取特定信息并从中取出特定的信息，通常需要配合 HttpClient 等请求库来获取 HTML 页面。
HttpClient 是用于发送 HTTP 请求和处理 HTTP 响应的工具库，并不具备 HTML 解析和操作功能。

什么是CompletableFuture，项目中如何使用它实现并发搜索?

CompletableFuture 是 Java 中用于支持异步编程和并发操作的类(在 JUC 并发包中)不仅可以通过CompletableFuture 处理异步任务，还能编排和构建复杂的异步操作流水线比如一个任务执行完后再触发另一个任务、等待任务都完成后再进行操作等。本项目中，我将用户搜索、文章搜索、图片搜索分别封装为

CompletableFuture 任务，然后使用 CompletableFuture.allOf组合这些任务、并发执行，并通过join 方法开启阻塞等待。等所有数据源搜索都完成后，才会执行后续的操作，返回数据给前端。

使用了门面模式来对各类数据源的搜索结果进行聚合，介绍门面模式的概念、作用和实现方式

门面模式的主要目的是提供一个 统一的 接口，用于访问一组子系统的功能。使用门面模式可以简化接口，降低调用方的使用和理解成本;它封装了子系统的复杂性，使客户端可以更轻松地与子系统交互，而不需要了解子系统内部的工作细节，降低了客户端和子系统的耦合度。
在本项目中，使用门面模式来对多个数据源的搜索结果进行聚合，让前端通过给同一个搜索接口传递不同的参数来灵活地获取数据。具体实现方式:
1.定义门面类:创建一个门面类，该类充当客户端和多个子系统之间的中介。门面类包含了对多个子系统的引用，并提供了封装接口，供客户端使用。
2.封装操作:在门面类中，根据输入参数，选择调用不同的数据源搜索服务来搜索数据，并将其结果进行合并、转换处理，以生成最终统一的聚合结果。
3.提供简化接口:前端通过与门面类交互来执行搜索操作，而不需要了解每个数据源具体的搜索过程。

你使用了适配器模式来实现新数据源的接入，请介绍适配器模式的概念，作用和实现方式

配器模式的主要目的是将一个类的接口转换成客户端所期望的另一个接口，使得原本由于接口不兼容而不能一起工作的类可以协同工作，就像是手机充电器的转接头一样。适配器模式的主要作用:
1.接口转换:适配器模式允许将一个类的接口转换成另一个类所期望的接口，使得两个类可以协同工作，而无需修改它们的源代码。
2.解耦合:适配器模式可以帮助解耦合不兼容的接口，使得客户端与适配器之间的接口保持一致，降低了代码的依赖性。
3.复用性:适配器模式可以将现有的类用于新的应用场景，增加了代码的复用性。
在本项目中，定制了统一的数据源接入规范(数据源接口)，比如任何接入聚合搜索系统的数据，必须要能够根据关键词搜索、并且支持分页搜索。在这个前提下，对于有些想要接入我们系统的数据源，如果原有的搜索方法参数和我们接口的定义不一致，又不能改造我们的接口以及对方原本的搜索方法，这时就需要使用适配器模式。比如搜索文章数据源，有一个现成的 searchFromEs 的方法，但接受的搜索参数是一个对象而不是 searchText 字符串，就需要使用适配器模式进行转换，对请求参数进行封装。

使用了注册器模式来管理多个数据源 对象，请介绍注册器模式的概念，作用和实现方式

注册器模式的主要目的是在应用程序中全局注册一些对象，便于被其他对象发现和使用，常用于管理和维护一组单例的全局对象。
使用注册器模式后，不仅能更方便地集中查找和获取全局对象，还避免了反复初始化的内存和时间开销。
具体的实现方式如下:
1.定义注册器类:将注册器类标识为一个 Bean，并且在类中添加一个 HashMap 属性，用于存放全局对象。
2.对象注册:通过 @PostConstruct 注解，在注册器 Bean 加载时初始化 HashMap 并且向其中插入新创建的数据源对象。
3.对象获取:提供一个根据 key(数据源类型)查找对象的方法，返回获取到的对象。

什么是代码的圈复杂度，为什么关注代码的圈复杂度?

圈复杂度是一种衡量软件代码复杂性的度量指标。它用于评估代码中的决策路径数量，即代码中分支语句(如 if、switch、while 等)的数量。

般来说，圈复杂度越高，代码的复杂性越高。
关注代码的圈复杂度，是为了消除代码中混乱的逻辑，让代码结构更清晰易懂，从而更好地维护代码。
在 IDEA IDE 中，可以使用 Metrics Reload 插件检测项目代码的圈复杂度，对于圈复杂度 >10 的代码，我会分析能否通过设计模式、抽象复用方法的方式降低复杂度，提高项目的代码质量。

什么是 Elasticsearch? Elasticsearch 和 MySQL 分别有哪些应用场景和优缺点?

Elasticsearch 是一个开源的分布式搜索引擎，提供了强大的全文搜索和复杂查询功能。
1)Elasticsearch
主要的应用场景:全文搜索、日志分析优点:搜索性能高、能够实现灵活的全文搜索、实时性强、支持分布式扩容缺点:部署运维成本较大，且查询操作学习成本较高
2)MySQL
主要的应用场景:关系型数据存储、事务和数据一致性支持(比如财务系统)
优点:支持事务、成熟稳定
缺点:实时搜索性能和灵活性较差，不利于大数据量的扩展
在实际业务中，可以结合使用这两种数据库，把需要全文检索功能的数据同步到 Elasticsearch上(比如文章数据);而其他无需检索的关系型数据(比如用户数据)存储在 MySQL中。

为什么ElasticSearch能实现更灵活的查询?

1)强大的查询DSL:Elasticsearch 提供了丰富的查询 DSL(领域特定语言)，允许用户以更高级别的抽象方式构建查询。DSL支持多种查询类型，包括全文搜索、精确配、范围查询、布尔查询、嵌套查询、通配符查询、模糊查询等。这些查询类型可以根据不同的搜索需求组合和嵌套，以构建复杂的查询条件。
2)全文搜索能力:Elasticsearch以全文搜索为基础，具有出色的文本搜索功能。它使用分析器和标记化技术来处理文本数据，支持词干分析、停用词过滤、同义词处理等，从而能够实现高效的文本搜索和相关性排名
3)倒排索引:Elasticsearch使用倒排索引来加速搜索。倒排索引是一种反向映射，将每个词汇与其出现在文档中的位置建立关联。这允许Elasticsearch在非常快的时间内查找到包含特定词汇的文档。
4)分布式架构:Elasticsearch采用分布式架构，数据分片和复制到多个节点上。这意味着查询可以并行执行，并且可以在多个节点上分散负载，以提高性能和可伸缩性。这对于处理大规模数据和高并发查询非常重要。
5)动态映射:Elasticsearch 具有动态映射功能，可以根据文档中的字段自动推断其数据类型。这意味着您可以灵活地插入文档，而不需要事先定义模式。这种灵活性允许您存储各种不同结构的文档，并以多种方式查询它们。
6)插件和定制性:Elasticsearch具有丰富的插件生态系统，可以轻松扩展其功能。您可以选择性地添加插件以满足特定需求，例如地理位置搜索、词汇推荐等。此外，Elasticsearch允许您自定义分析器、标记化器和过滤器，以适应不同的数据类型和语言。

你提到采用动静锋利的策略存储文章信息，请解释一下动静分离的概念，以及它在项目中具体的实现方式?

动静分离是一种常见的架构设计策略，是指将一个应用程序的动态数据(通常是经常变化的数据，如点赞数)与静态数据(通常是不经常变化的数据，如文章正文)分别存储在不同的存储引擎(或表)中，从而优化性能、降低负载等。本项目中，我将文章的内容(静态数据)存储到 Elasticsearch 中用于实现检索，而文章的点赞数等动态数据存储到 MySQL 中。
在用户根据关键词搜索文童时，会先从 Elasticsearch 分词搜索出文章的 id 获取到静态数据然后根据 id 在数据库内查找对应的文章，从而获取到最新的动态数据。这样做可以减轻 MySQL的负载，同时实现快速的全文搜索和数据展示。

请介绍Kibana的作用和它的基本功能，你在项目中使用了Kibana的哪些功能

Kibana 是一个开源的数据可视化平台，主要用于对 Elasticsearch 中的数据进行搜索、分析和可视化展示。基本功能包括:数据查询搜索、数据可视化看板、日志分析、Elasticsearch 操作调试在项目中，我在 Kibana 上搭建了 Elasticsearch 文章总数看板，用于观测 MySQL 成功同步到Elasticsearch 的数据量。此外，在开发 Elasticsearch 文章搜索功能时，我使用 Kibana的DevTools 编写 DSL 代码来调试 Elasticsearch 的复杂查询条件，提高了开发效率。

由于 Elasticsearch 的查询语法复杂多样，为了保证业务代码中指定的搜索条件的正确性，我先把可能的搜索条件以 DSL 的方式在 Kibana DevTools 中执行验证。掌握了 Elasticsearch 的常用操作语法后，再去编写代码，提高了整体的开发效率。Kibana DevTools 提供了一个界面，可以通过编写 DSL 代码来操作 Elasticsearch。

Spring Data ElasticSearch的QueryBuilder

QueryBuilder 是 Spring Data Elasticsearch 提供的、用于构建复杂查询条件的工具(有点类似MyBatis Plus的QueryWrapper).
在项目中，我使用 boolQueryBuilder 组合多个子查询条件，比如同时根据 userId 和 id 来过滤;使用 matchQuery 实现全文检索，比如根据关键词搜索文章内容;使用 termQuery 实现精确匹配查询，比如根据标签搜索文章。

在MYSQL和ElasticSearch的数据同步过程中，如何解决重复更新和插入的问题

1)使用唯一标识符:同一篇文章数据，在 MySQL和 Elasticsearch 中的 id(索引)是相同且唯一的。在同步数据时，可以使用主键来确保每个文档只被索引一次;如果文档已存在于Elasticsearch 中，可以使用主键来更新它，否则将其插入为新文档。
2)确保同步操作幂等:即多次执行相同的操作不会产生不同的结果，即使重复执行同步操作结果也应该和不重复执行保持一致。
3)数据变更监听:使用 Canal同步数据，保证每个数据库变更事件只捕获并处理一次，如果处理失败，通过日志或者 DB 记录异常信息，通过补偿机制回写数据。

什么是Canal，它有什么作用，请简述它的核心实现原理?

Canal 是阿里开源的数据库日志监听工具，主要用途是基于 MySQL 数据库增量日志解析，及时捕获数据库的变更操作并传递给下游应用，下游应用可以根据数据库的变更实现各种操作，比如数据同步、缓存更新、实时分析等。建议阅读官方文档:https://github.com/alibaba/canal
Canal 的核心实现原理:
·canal 模拟 MySQL slave 的交互协议，伪装自己为 MySQL slave，向 MySQL master 发送dump 协议
·MySQL master 收到 dump 请求，开始推送 binary log 给 slave (即 canal)
canal 解析 binary log 对象(原始为 byte 流)

你在项目中使用了 Swagger+ Knife4j自动生成接口文档，请谈谈 Swagger 和 Knife4j的作用和它们对项目开发的影响。

Swagger 是一个用于自动构建和生成可交互接口文档的工具集。使用 Swagger 接口文档生成工具后，我不需要在开发完项目后手动编写一套接口文档，而是直接交由系统自动根据Controller 接口层的代码自动生成文档，大幅节省时间。使用 Swagger 生成的接口文档不仅能够分组查看请求参数和响应，还支持灵活的在线调试，可以直接通过界面发送请求来测试接口，提高开发调试效率。此外，引入 Swagger后，可以得到

基于 OpenAPI 规范的接口定义 JSON，可以配合第三方工具来根据 JSON 自动生成前端请求代码、自动生成客户端调用 SDK 等。Knife4j是 swagger 的增强版，能够生成更美观的 API 接口文档，并且提供了离线文档导出.接口分组排序等增强功能。(参考官网:https://doc.xiaominfo.com/docs/features)

项目是否有上线，如何实现前端页面部署的

项目有实际上线。我是通过本地打包 +Nginx 实现了前端页面部署
具体过程如下:
1.购买云服务器
2.安装和初始化宝塔 Linux 面板，会自动安装 Nginx 服务器
3.在宝塔上创建一个网站
4.本地使用 npm run build 命令打包项目，得到 dist 网站静态文件目录
5.上传本地打包好的 dist 目录到服务器，然后配置 Nginx 指向文件目录路径，即可访问前端静态文件
还有其他的部署方式，比如使用 Vercel等 Serverless 服务一键部署到第三方托管服务器，但由于本人有服务器、并且想实践下 Nginx 配置，所以没有选择这种方式。

介绍一下项目的完整业务流程

用户可以在前端通过一个搜索框集中搜索出不同来源的数据，比如通过离线爬虫提前获取到的文章数据、通过 Bing 接口实时获取到的图片数据、以及业务内部数据库的用户数据。