1. 对比书上各种软件体系结构风格和视图特点，思考自己项目属于哪种设计风格？

答：属于客户端-服务器架构，这是一种常见的体系结构，其中客户端应用程序通过网络请求与服务器通信。对于人才招聘网站，客户端可以是用户使用的网页浏览器，而服务器则负责处理用户请求并提供相应的数据和功能。这种风格有助于将业务逻辑与用户界面分离，使得系统更易于维护和扩展。

1. 参阅课本和网上资料，研究经典软件体系结构案例KWIC。

KWIC（关键词在上下文中的索引）是一个经典的软件体系结构案例，用于文本处理和信息检索。它的主要目标是提供一种快速有效的方法来检索文本中特定关键词的上下文信息。KWIC系统的典型架构包括以下几个主要组件：

1. 输入模块：负责接收文本输入，并将其传递给下一个处理阶段。

2. 预处理模块：这一阶段对输入文本进行预处理，通常包括去除停用词、标点符号、进行大小写统一等操作，以便更好地处理和索引。

3. 索引模块：主要任务是将文本中的关键词与其上下文信息建立索引，以便后续的检索。通常使用数据结构如排序数组、哈希表或树结构来实现高效的索引。

4. 检索模块：接受用户查询，并根据索引提供相关的上下文信息。这可能涉及到搜索、排序和展示匹配结果等操作。

5. 输出模块：负责将检索到的结果以适当的形式呈现给用户，通常是按照特定格式或排版的文本。

KWIC系统的架构设计旨在实现高效的文本检索和呈现，以满足用户对信息的快速访问需求。

分层模式（Layered Architecture）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性 | 优先级 | 共享数据 | 数据抽象 | 隐含调用 | 管道和过滤器 |
| 易于改变算法 | 2 | 2 | 4 | 4 | 5 |
| 易于改变数据表示 | 5 | 2 | 5 | 4 | 1 |
| 易于改变功能 | 3 | 4 | 4 | 3 | 5 |
| 好的性能 | 3 | 4 | 2 | 3 | 5 |
| 有效的数据表示 | 4 | 5 | 5 | 5 | 2 |
| 易于复用 | 5 | 1 | 4 | 5 | 4 |
| 总计 |  | 63 | 91 | 91 | 73 |

客户端-服务器架构（Client-Server）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性 | 优先级 | 共享数据 | 数据抽象 | 隐含调用 | 管道和过滤器 |
| 易于改变算法 | 2 | 2 | 4 | 4 | 5 |
| 易于改变数据表示 | 5 | 2 | 5 | 4 | 1 |
| 易于改变功能 | 3 | 4 | 4 | 3 | 5 |
| 好的性能 | 3 | 4 | 2 | 3 | 5 |
| 有效的数据表示 | 4 | 5 | 4 | 5 | 3 |
| 易于复用 | 5 | 3 | 4 | 5 | 4 |
| 总计 |  | 73 | 86 | 91 | 77 |

最佳为客户端-服务器架构。

1. 项目的SAD

**软件架构设计文档 (SAD) - 网上人才招聘网站（初稿）**

**1. 引言**

本文档旨在描述网上人才招聘网站的软件架构设计。该网站旨在为求职者和招聘企业提供一个平台，使其能够发布和搜索工作机会，并进行招聘流程管理。

**2. 系统概述**

网上人才招聘网站将包括以下主要模块：

用户管理

职位管理

简历管理

搜索和匹配

消息通知

系统管理

**3. 架构设计**

3.1 技术栈

后端：Java 11, Spring Boot, Hibernate

前端：React.js, HTML, CSS, JavaScript

数据库：MySQL

消息队列：Kafka

搜索引擎：Elasticsearch

容器化：Docker, Kubernetes

3.2 架构风格

采用微服务架构，将系统拆分为多个服务：

用户服务

职位服务

简历服务

消息服务

搜索服务

系统管理服务

3.3 数据库设计

采用关系型数据库MySQL存储数据，分为用户信息表、职位信息表、简历信息表等。

3.4 安全设计

使用JWT实现用户认证和授权，对敏感信息进行加密存储，并实现访问控制列表（ACL）。

3.5 部署架构

采用Docker容器化部署，使用Kubernetes进行集群管理，实现高可用和水平扩展。

3.6 性能设计

通过负载均衡、缓存和异步处理提高系统性能，使用Elasticsearch实现快速搜索和匹配。

**4. 系统交互**

4.1 用户界面

前端采用React.js框架，实现响应式设计，提供用户友好的交互界面。

4.2 API设计

采用RESTful API设计风格，提供统一的接口规范，支持前后端分离开发。

**5. 非功能性需求**

5.1 可扩展性

系统应支持水平扩展，能够方便地增加新的功能模块和服务。

5.2 可维护性

代码应遵循规范，具有良好的可读性和可维护性，便于团队协作开发和维护。

5.3 安全性

系统应具备高安全性，确保用户信息和数据的保密性和完整性。

5.4 性能

系统应具备良好的性能，能够快速响应用户请求并支持大并发访问。

**6. 总结**

本文档描述了网上人才招聘网站的软件架构设计，包括技术栈、架构风格、数据库设计、安全设计、部署架构、系统交互和非功能性需求。这些设计将为项目的实施和开发提供指导，并确保系统具备良好的可扩展性、可维护性、安全性和性能