清云ClearCloud短视频

详

细

设

计

参赛团队：铁道游击队

团队成员：罗义恒 吴静 彭政

2023 年 11 月 6 日

目录

[**1．引言** 2](#_Toc150293682)

[**1.1 专业术语** 2](#_Toc150293683)

[**1.2 项目分工** 3](#_Toc150293684)

[**2．总体设计** 4](#_Toc150293685)

[**2.1 功能结构** 4](#_Toc150293686)

[**2.2 技术选型** 4](#_Toc150293687)

[**3．架构设计** 5](#_Toc150293688)

[**3.1 架构分层** 5](#_Toc150293689)

[**3.2 代码架构** 6](#_Toc150293690)

[**4. 核心功能设计** 7](#_Toc150293691)

[**4.1 上传视频** 7](#_Toc150293692)

[**4.2 拉取视频流** 8](#_Toc150293693)

[**5. 数据库设计** 8](#_Toc150293694)

[**5.1 关系型数据库设计** 8](#_Toc150293695)

[**5.2 非关系型数据库设计** 9](#_Toc150293696)

[**6.项目亮点** 10](#_Toc150293697)

[**6.1 使用Redisson的布隆过滤器对视频流去重** 10](#_Toc150293698)

[**6.2 Redis优先采用批量读取** 10](#_Toc150293699)

[**6.3使用国密SM3加密用户隐私数据** 10](#_Toc150293700)

[**6.4使用双token无感刷新** 11](#_Toc150293701)

**1．引言**

**1.1 专业术语**

* SpringBoot 开发框架：SpringBoot是一个快速开发、便于配置、基于Spring的Java开发框架。
* Redis数据库：Redis是一个基于内存的开源键值存储系统，用于快速存储和访问数据。
* MySQL数据库：MySQL是一种开源的关系型数据库管理系统，广泛用于存储、管理和检索大量结构化数据，并提供高性能和可靠性的数据存储解决方案。
* Vue：Vue是一款流行的JavaScript前端框架，用于构建响应式的、可组件化的用户界面。
* Nacos：Nacos是一个开源的动态服务发现、配置管理和服务管理平台，用于构建和管理云原生应用。
* SpringCloud Gateway：Spring Cloud Gateway是基于Spring Framework的轻量级网关服务，用于构建和管理微服务架构中的API网关，提供路由、过滤、负载均衡等功能，实现统一的访问控制和请求转发。
* OpenFeign：OpenFeign是一个基于注解的声明式HTTP客户端库，简化了微服务架构中服务间调用的代码编写和配置工作。
* Redisson：Redisson是一个基于Redis的分布式Java对象和服务的框架，提供了丰富的功能和API，简化了在分布式环境中的数据操作和并发控制。
* JWT：JWT（JSON Web Token）是一种基于JSON格式的开放标准，用于在网络应用间安全地传输信息，通过数字签名保证数据的完整性和信任，常用于身份验证和授权机制。
* 七牛云Kodo：七牛云海量存储系统（Kodo）是自主研发的非结构化数据存储管理平台，支持中心和边缘存储。 平台经过多年大规模用户验证已跻身先进技术行列，并广泛应用于海量数据管理的各类场景。
* Swagger: Swagger是一个开源的规范和工具集，用于设计、构建、文档化和测试RESTful API，简化API开发过程并提供交互式文档和代码生成等功能。

**1.2 项目分工**

|  |  |
| --- | --- |
| **团队成员** | **主要贡献** |
| 罗义恒 | 后端技术选型，数据库设计，后端开发，文档编写。 |
| 吴静 | 前端技术选型，前端开发，视频录制 |
| 彭政 | 产品经理 |

**2．总体设计**

**2.1 功能结构**

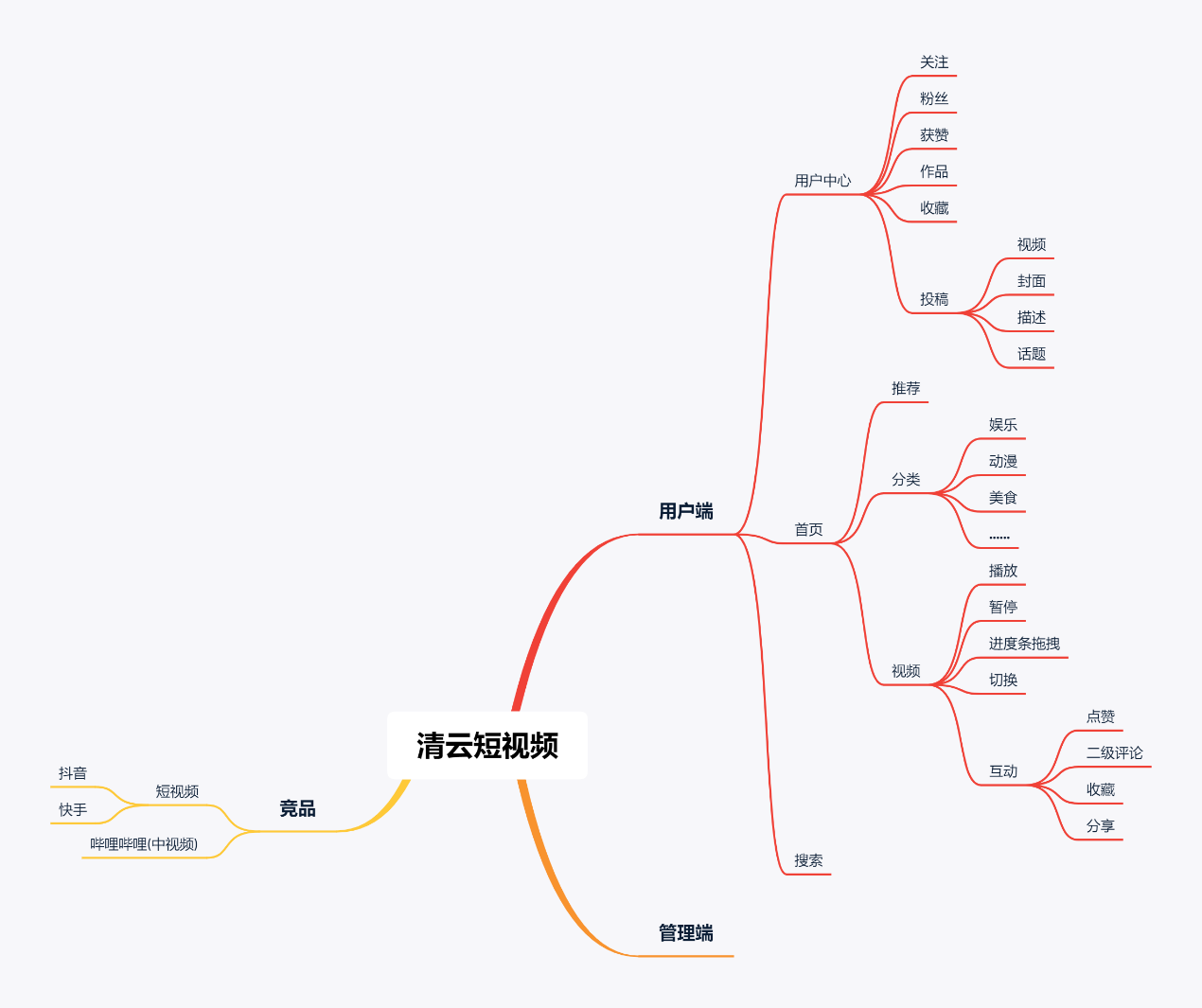


图2-1 功能结构图

**2.2 技术选型**

* 工具：IntelliJ IDEA + Visual Studio Code
* 数据库：MySQL 8.0.24 + Redis 7.0.5
* 开发语言：Java JDK17 + JavaScript + CSS3 + HTML5
* 开发框架：SpringBoot2.7.17 + Mybatis-Plus3.5.2 + Vue3
* 服务注册与发现：Nacos2.1.0
* 网关：Spring Cloud Gateway
* 服务通信：OpenFeign
* 鉴权：JWT
* 文件存储：七牛云Kodo对象存储
* 项目管理：Git + Docker

**3．架构设计**

**3.1 架构分层**

清云ClearCloud短视频应用分为七层，分别是：

* 应用端层：用户的访问途径。
* 前端UI层：负责与用户交互的前端界面，包括用户界面的设计、逻辑控制和数据呈现。前端UI层通常运行在用户的设备上，如PC、移动设备等，与后端系统通过网络进行数据交互。
* 网络层：负责处理视频等数据传输和通信。
* 访问控制层：负责验证和控制系统用户访问权限的组件。访问控制层可以根据用户的身份和角色，限制用户访问系统的资源和功能，以确保系统的安全和稳定性。
* 服务层：负责提供数据和业务逻辑处理，服务层包括多个服务模块，每个模块负责不同的业务功能，通过API接口提供给其他组件或外部系统调用。
* 存储层：负责存储清云短视频产生的中间数据、结果数据，主要使用关系数据库 MySQL、非关系数据库 Redis以及Kodo文件存储服务。
* 基础设施层：提供支持和服务于上层组件的基础设施组件。

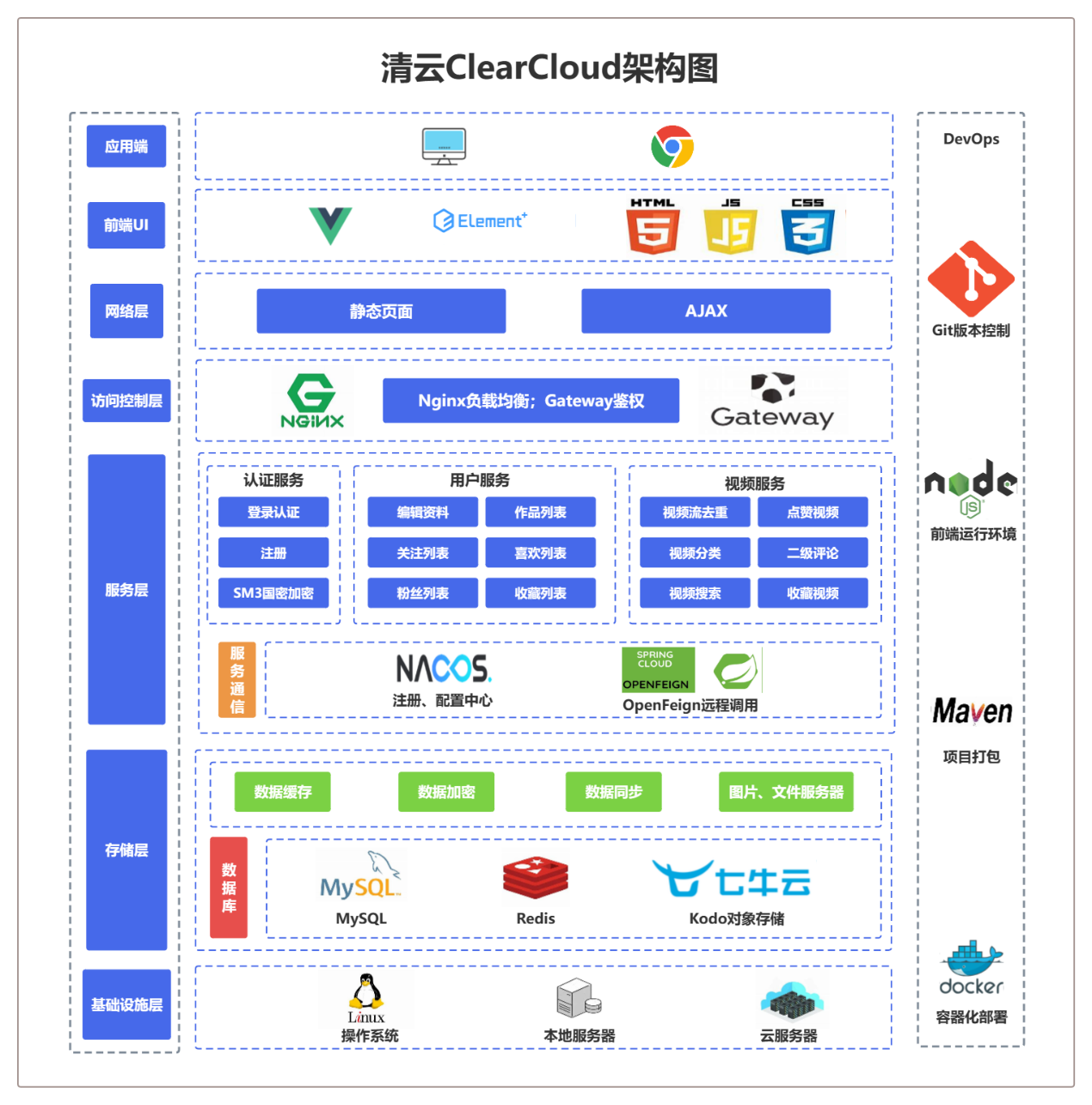


图 3 系统架构图

**3.2 代码架构**

**项目整体的代码框架**



* Parent模块：负责同一管理依赖版本
* Feign-api模块：抽取所有需要远程调用的api
* Gateway模块：负责负载均衡以及鉴权校验
* User-service模块：负责提供与用户密切相关的服务
* Video-service模块：负责提供与视频处理密切相关的服务

**4. 核心功能设计**

**4.1 上传视频**

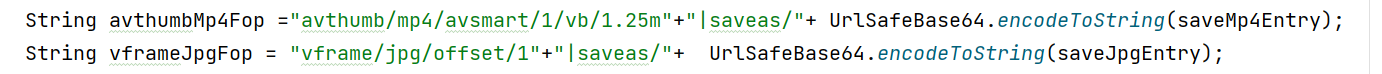
**[功能说明]**

用户可以投稿自己创作的视频

**[处理]**

（1）清云短视频使用七牛玉kodo对象存储，采用服务端直传，异步返回结果。

（2）在上传token 中设置 转码命令(avthumb)和视频截帧命令(vframe)



（3）返回上传视频的URL和视频封面URL(视频第一帧)

**4.2 拉取视频流**

**[功能说明]**

拉取视频流推送给用户

**[处理]**

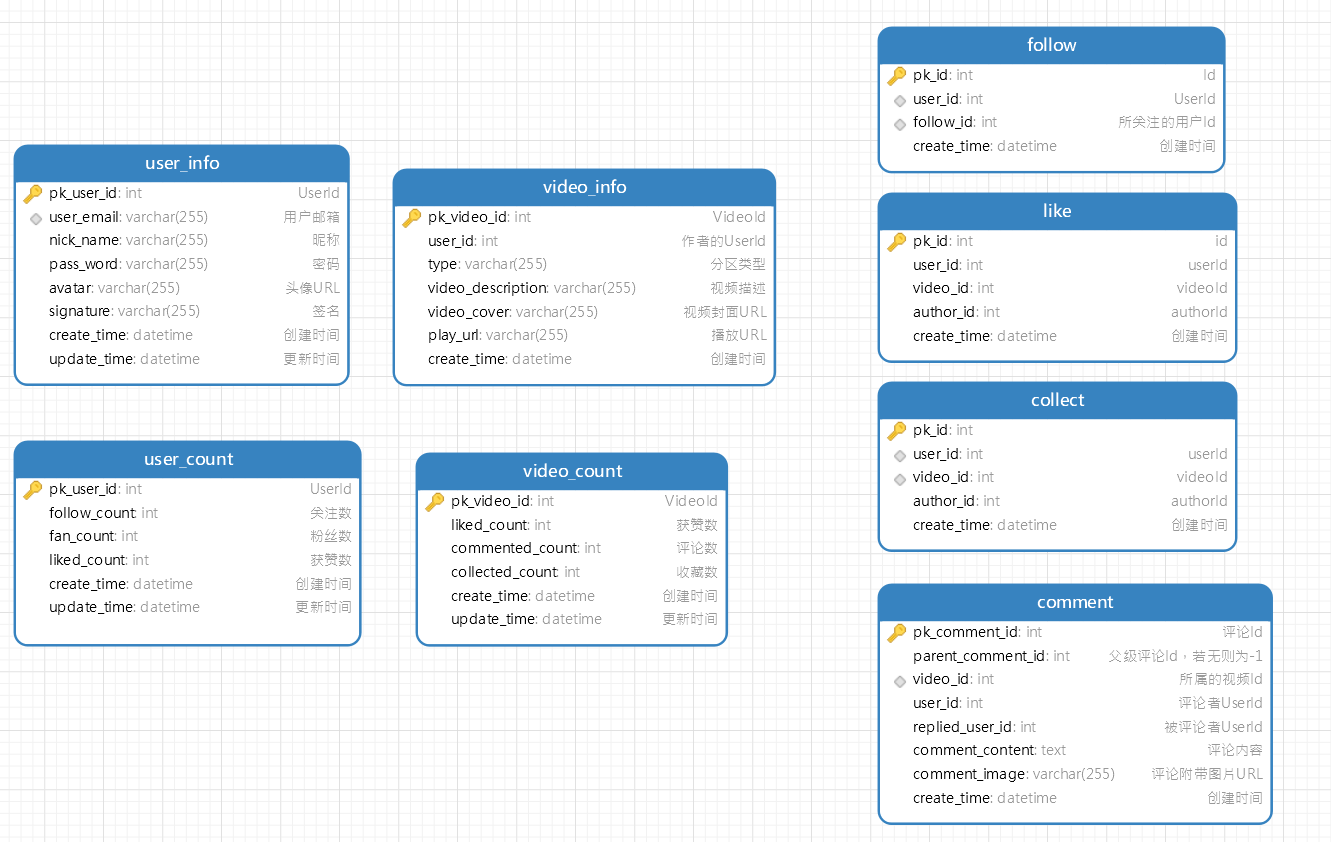
（1）判断用户是否分区，如果用户选择了分区则按类别初步检索出相关视频

（2）用户在注册时，使用redisson为每个用户创建一个布隆过滤器，记录用户看过的视频，使用布隆过滤器过滤掉用户观看过的视频

（3）返回过滤后的视频流给用户

**5. 数据库设计**

**5.1 关系型数据库设计**



* 索引
* user\_info: user\_email唯一索引
* video\_info: user\_id 普通索引
* follow: （user\_id, follow\_id)联合索引
* collect: （user\_id, video\_id)联合索引
* comment: video\_id普通索引

**User服务**

**对整个 user\_info 进行了垂直分表，一个是user的基本静态信息，一个是 user\_count，存储user的计数信息**，这样做有助于简化数据库结构，**防止部分信息的经常修改导致整个缓存经常失效**，提高查询性能，并支持数据的维护和扩展。

**Video服务**

**与User服务同理，对整个 video\_info 进行了垂直分表，一个是video的基本静态信息，一个是 video\_count，存储video的计数信息**。

**5.2 非关系型数据库设计**

清云ClearCloud短视频系统使用Redis作为非关系数据库。Redis可以提供高速的数据读写，负责处理热点数据或临时数据的增删查改业务。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 服务 | 数据信息 | 数据类型 | **key** | **value** | 缓存策略 |
| **user** | user\_info | String | ClearCloud:UserInfo: user\_id | user\_info  静态信息 | 旁路缓存 |
| **user** | user\_count | String | ClearCloud:UserCount: user\_id | user\_count  计数信息 | 写回 |
| **user** | 关注用户的id | set | ClearCloud:Follow: user\_id | 关注用户的ids | 旁路缓存 |
| **user** | 用户点赞的视频id | set | ClearCloud:Like: user\_id | 点赞的视频ids | 写回 |
| **video** | video\_count | String | ClearCloud:VideoCount:video\_id | Video的计数信息 | 写回 |
| **video** | 系统发布的视频 id | zset | ClearCloud: PublishedVideo | Score: 发布时间戳  member: video\_id |  |

**6.项目亮点**

**6.1 使用Redisson的布隆过滤器对视频流去重**

用户在注册时，使用redisson为每个用户创建一个布隆过滤器，记录用户看过的视频，使用布隆过滤器过滤掉用户观看过的视频。

1. 高效的去重能力：Redisson的布隆过滤器是基于Redis的实现，它使用了高效的位数组和哈希函数，可以非常快速地判断一个元素是否存在于过滤器中。对于大规模的视频流数据，它能够高效地进行去重操作，减少重复视频的存储和传输。
2. 内存占用低：布隆过滤器使用了紧凑的位数组结构，相对于传统的数据结构，它所需的内存占用非常低。对于视频流去重的场景，这意味着可以使用较小的内存空间来处理大量的视频数据。
3. 兼容性和易用性：Redisson是一个流行的Java Redis客户端库，它提供了对Redis的全面支持，并且具有简单易用的API。使用Redisson的布隆过滤器，可以方便地与其他Redis功能和数据结构集成，例如使用Redis的持久化功能来保存过滤器的状态。

**6.2 Redis优先采用批量读取**

1. 减少网络开销：与多次单独的读取请求相比，批量读取可以减少网络通信的开销。在一次请求中获取多个键值对，可以降低网络延迟并提高读取效率，特别是在跨网络或高延迟环境中。
2. 提高系统性能：批量读取可以减少与Redis服务器的交互次数，从而减轻服务器的负载。通过一次批量读取请求，可以获取多个键值对的数据，减少了服务器的处理时间和资源消耗，提高了系统的整体性能。
3. 降低客户端开销：对于客户端应用程序来说，多次单独的读取请求会增加连接和协议开销。通过批量读取，可以将多个键值对的读取合并为一个请求，减少了连接建立和数据包头部的开销，降低了客户端的开销。

**6.3使用国密SM3加密用户隐私数据**

1. 安全性高：国密SM3是中国密码学标准中的哈希算法，具有较高的安全性和抗攻击能力。它采用了非对称结构和混合置换运算，能够有效地抵御碰撞攻击、预映像攻击等密码学攻击手段。使用SM3加密用户隐私信息可以保护用户数据的机密性和完整性。
2. 合规要求：在某些国家或行业中，使用特定的密码算法进行数据保护是法律或合规要求。对于中国地区或与中国相关的应用场景，使用国密SM3加密用户隐私信息符合相关的法规和标准，能够满足数据安全合规的要求。
3. 技术支持和生态系统：国密SM3算法在中国得到广泛的支持和应用，有成熟的技术支持和丰富的生态系统。相关的密码学库、工具和文档资源丰富，可以方便地集成和使用SM3算法，加速系统开发和部署。

**6.4使用双token无感刷新**

1. 提高用户体验：传统的单 Token 刷新机制需要用户重新输入用户名和密码或进行额外的认证步骤才能获取新的访问令牌（Access Token）。而双 Token 无感刷新机制可以在用户不知情的情况下自动刷新访问令牌，无需用户的干预，从而提高了用户的使用体验。
2. 增强安全性：双 Token 无感刷新机制通过使用不同的令牌类型和过期时间来增强安全性。通常，访问令牌的过期时间较短，而刷新令牌（Refresh Token）的过期时间较长。当访问令牌过期时，应用程序可以使用刷新令牌获取新的访问令牌，而无需存储用户的用户名和密码。这有助于减少访问令牌的泄露风险，提高系统的安全性。
3. 减少令牌传输次数：使用双 Token 无感刷新机制，应用程序可以在访问令牌过期之前通过刷新令牌自动获取新的访问令牌。这意味着应用程序可以减少向身份验证服务器发送的令牌刷新请求，降低了网络开销和服务器负载，提高了系统的性能和可扩展性。