

# 乐拍视界项目文档

项目小组 第 2 小组

小组成员 柏翔 刘金林 兰寅银 朱灿银 周俊

联系方式 17748752006

重庆师范大学软件工程系

## 摘要

“乐拍视界”项目旨在解决年轻群体创作音乐短视频时操作繁琐、社交孤立的痛点，通过打造一个集智能拍摄、海量配乐、一键美化与沉浸式社交于一体的移动平台，开创短视频新赛道。项目计划以 6-12 个月完成开发，核心是整合音乐库、智能算法与推荐系统，旨在通过极致流畅的一体化体验，快速吸引用户并构建活跃社区，最终成为引领年轻文化的领先平台。

# 目录

|                            |           |
|----------------------------|-----------|
| <b>摘要.....</b>             | <b>2</b>  |
| <b>第1章 立项.....</b>         | <b>10</b> |
| 1.1. 项目起源与提案.....          | 10        |
| 1. 发现问题.....               | 10        |
| 2. 根源分析.....               | 10        |
| 3. 系统边界定义.....             | 10        |
| 4. 约束确定.....               | 10        |
| 5. 提案构想.....               | 10        |
| 1.2. Business Case.....    | 11        |
| 1. 摘要.....                 | 11        |
| 2. 市场机遇.....               | 11        |
| 3. 目标市场与客户细分.....          | 12        |
| 4. 竞争优势.....               | 12        |
| 5. 市场营销与用户获取策略.....        | 12        |
| 6. 风险与应对.....              | 12        |
| 7. 成本估算.....               | 13        |
| 8. 项目目标.....               | 13        |
| <b>第2章 愿景.....</b>         | <b>14</b> |
| 2.1. 问题陈述.....             | 14        |
| 1. 问题一.....                | 14        |
| 2. 问题二.....                | 14        |
| 3. 问题三.....                | 15        |
| 2.2. 涉众与用户.....            | 15        |
| 1. 涉众.....                 | 15        |
| 2. 用户.....                 | 17        |
| 2.3. 关键涉众和用户的需要.....       | 17        |
| 2.4. 产品概述.....             | 20        |
| 1. 产品定位陈述.....             | 20        |
| 2. 完整的产品概述.....            | 20        |
| 2.1. 能力概述.....             | 20        |
| 2.2. 客户效益 (Benefits) ..... | 21        |
| 2.3. 假设和依赖.....            | 21        |
| 2.4. 取舍和竞争.....            | 22        |
| 2.5. 产品特性.....             | 22        |
| 1. 内容消费者.....              | 22        |

|                          |           |
|--------------------------|-----------|
| 2. 内容创作者.....            | 23        |
| 3. 管理类.....              | 24        |
| 4. 业务处理类.....            | 25        |
| 4.1. 审核人员.....           | 25        |
| 4.2. 客服.....             | 26        |
| 2.6. 其他产品需求.....         | 26        |
| 2.7. 特性优先级.....          | 27        |
| 2.8. 补充说明.....           | 29        |
| 1. 法律 / 规定性需求.....       | 29        |
| 2. 应用开发标准.....           | 29        |
| 3. 质量属性需求.....           | 30        |
| 4. 设计与实现约束.....          | 30        |
| 5. 其他补充需求.....           | 30        |
| <b>第3章 用况建模.....</b>     | <b>31</b> |
| 3.1. 术语表.....            | 31        |
| 3.2. 参与者清单.....          | 32        |
| 1. 识别主参与者.....           | 32        |
| 2. 参与者简要描述.....          | 32        |
| 2.1. 普通用户.....           | 32        |
| 2.2. 内容审核员.....          | 32        |
| 2.3. 客服.....             | 32        |
| 2.4. 运营人员.....           | 32        |
| 3.3. 乐拍视界的普通用户主要用况.....  | 32        |
| 1. 浏览个性化视频流用况.....       | 33        |
| 1.1. 简要描述.....           | 33        |
| 1.2. 前置条件.....           | 33        |
| 1.3. 后置条件.....           | 33        |
| 1.4. 基本事件流.....          | 33        |
| 1.5. 备选流.....            | 34        |
| 3.4. 乐拍视界的内容审核员主要用况..... | 36        |
| 1. 审核用户发布内容用况.....       | 37        |
| 1.1. 简要描述.....           | 37        |
| 1.2. 前置条件.....           | 37        |
| 1.3. 后置条件.....           | 37        |
| 1.4. 基本事件流.....          | 37        |
| 1.5. 备选流.....            | 38        |
| 2. 管理内容审核规则用况.....       | 39        |

|                         |    |
|-------------------------|----|
| 2.1. 简要描述.....          | 39 |
| 2.2. 前置条件.....          | 39 |
| 2.3. 后置条件.....          | 39 |
| 2.4. 基本事件流.....         | 39 |
| 2.5. 备选流.....           | 40 |
| 3.5. 乐拍视界的客服主要用况.....   | 41 |
| 1. 处理用户咨询与投诉用况.....     | 41 |
| 1.1. 简要描述.....          | 41 |
| 1.2. 前置条件.....          | 41 |
| 1.3. 后置条件.....          | 41 |
| 1.4. 基本事件流.....         | 41 |
| 1.5. 备选流.....           | 42 |
| 2. 管理客服知识库用况.....       | 43 |
| 2.1. 简要描述.....          | 43 |
| 2.2. 前置条件.....          | 43 |
| 2.3. 后置条件.....          | 43 |
| 2.4. 基本事件流.....         | 43 |
| 2.5. 备选流.....           | 44 |
| 3.6. 乐拍视界的运营人员主要用况..... | 45 |
| 1. 策划并执行运营活动用况.....     | 45 |
| 1.1. 简要描述.....          | 45 |
| 1.2. 前置条件.....          | 45 |
| 1.3. 后置条件.....          | 46 |
| 1.4. 基本事件流.....         | 46 |
| 1.5. 备选流.....           | 46 |
| 2. 管理内容推荐策略用况.....      | 47 |
| 2.1. 简要描述.....          | 47 |
| 2.2. 前置条件.....          | 47 |
| 2.3. 后置条件.....          | 47 |
| 2.4. 基本事件流.....         | 47 |
| 2.5. 备选流.....           | 48 |
| 3. 监控平台运营数据用况.....      | 49 |
| 3.1. 简要描述.....          | 49 |
| 3.2. 前置条件.....          | 49 |
| 3.3. 后置条件.....          | 49 |
| 3.4. 基本事件流.....         | 49 |
| 3.5. 备选流.....           | 50 |

|  |           |
|--|-----------|
| <b>第4章 需求分析.....</b>                                   | <b>51</b> |
| <b>4.1. 分析模型概述.....</b>                                | <b>51</b> |
| <b>1. 分析模型的目标与关注点.....</b>                             | <b>51</b> |
| <b>2. 分析模型的组成.....</b>                                 | <b>51</b> |
| <b>3. 用况的实现 (Realization) .....</b>                    | <b>51</b> |
| <b>4.2. 健壮性分析.....</b>                                 | <b>52</b> |
| <b>1. 边界类.....</b>                                     | <b>52</b> |
| <b>2. 实体类.....</b>                                     | <b>53</b> |
| <b>3. 控制类.....</b>                                     | <b>53</b> |
| <b>4.3. 交互建模.....</b>                                  | <b>54</b> |
| <b>1. 浏览个性化视频.....</b>                                 | <b>54</b> |
| <b>1.1. 参与对象 (协作角色) .....</b>                          | <b>54</b> |
| <b>1.2. 通讯图.....</b>                                   | <b>55</b> |
| <b>2. 审核视频.....</b>                                    | <b>56</b> |
| <b>2.1. 参与对象 (协作角色) .....</b>                          | <b>56</b> |
| <b>2.2. 通讯图.....</b>                                   | <b>56</b> |
| <b>3. 创建发布活动.....</b>                                  | <b>57</b> |
| <b>3.1. 参与对象 (协作角色) .....</b>                          | <b>57</b> |
| <b>3.2. 通讯图.....</b>                                   | <b>57</b> |
| <b>4.4. 职责分配与 CRC 卡.....</b>                           | <b>58</b> |
| <b>1. 浏览个性化视频相关类.....</b>                              | <b>58</b> |
| <b>1.1. Class Name: FeedController (控制类).....</b>      | <b>58</b> |
| <b>1.2. Class Name: RecommendationModel (实体类).....</b> | <b>58</b> |
| <b>1.3. Class Name: Video (实体类).....</b>               | <b>59</b> |
| <b>1.4. Class Name: User (实体类).....</b>                | <b>59</b> |
| <b>2. 审核用户发布内容相关类.....</b>                             | <b>59</b> |
| <b>2.1. Class Name: AuditController (控制类).....</b>     | <b>59</b> |
| <b>2.2. Class Name: AuditTask (实体类).....</b>           | <b>59</b> |
| <b>3. 策划并执行运营活动相关类.....</b>                            | <b>60</b> |
| <b>3.1. Class Name: ActivityController (控制类).....</b>  | <b>60</b> |
| <b>3.2. Class Name: OperationActivity (实体类).....</b>   | <b>60</b> |
| <b>4.5. 整合分析类图.....</b>                                | <b>60</b> |
| <b>1. 实体类 (Entity Classes).....</b>                    | <b>60</b> |
| <b>1.1. Video (短视频).....</b>                           | <b>60</b> |
| <b>1.2. User (用户).....</b>                             | <b>61</b> |
| <b>1.3. Music (音乐).....</b>                            | <b>61</b> |
| <b>1.4. AuditTask (审核任务).....</b>                      | <b>61</b> |
| <b>1.5. OperationActivity (运营活动).....</b>              | <b>62</b> |

|                                |           |
|--------------------------------|-----------|
| 2. 控制类 (Control Classes).....  | 62        |
| 3. 边界类 (Boundary Classes)..... | 63        |
| 4. 类关联关系.....                  | 63        |
| 4.1. 用户与视频 (发布关系).....         | 63        |
| 4.2. 视频与音乐 (使用关系).....         | 63        |
| 4.3. 审核任务与视频 (审核对象关系).....     | 64        |
| 4.4. 审核任务与用户 (执行关系).....       | 64        |
| 4.5. 运营活动与视频 (聚合关系).....       | 65        |
| <b>第5章 架构设计.....</b>           | <b>66</b> |
| <b>5.1. 设定权衡优先级.....</b>       | <b>66</b> |
| 1. 核心设计目标的优先级排序.....           | 66        |
| 2. 具体的架构权衡决策.....              | 67        |
| 2.1. 空间换时间.....                | 67        |
| 2.2. 可用性优于一致性.....             | 67        |
| 2.3. 功能分级与快速迭代.....            | 67        |
| 2.4. 动态性优于静态效率.....            | 67        |
| 3. 架构决策的验证与调整.....             | 68        |
| <b>5.2. 估算系统性能.....</b>        | <b>68</b> |
| 1. 基础假设与模型参数.....              | 68        |
| 2. 计算吞吐量.....                  | 68        |
| 2.1. 视频浏览.....                 | 68        |
| 2.2. 视频上传.....                 | 69        |
| 2.3. 点赞视频.....                 | 69        |
| 3. 估算存储需求.....                 | 69        |
| 4. 估算网络带宽.....                 | 69        |
| 5. 性能估算总结与架构调整.....            | 70        |
| <b>5.3. 选择架构风格.....</b>        | <b>70</b> |
| 1. 整体系统架构：分布式客户/服务器风格.....     | 70        |
| 2. 客户端架构：MVC 风格.....           | 71        |
| 3. 服务器端架构：分层与分区结合.....         | 71        |
| 4. 视频处理子系统：管道-过滤器风格.....       | 72        |
| 5. 数据存储架构：主从复制与读写分离.....       | 72        |
| <b>5.4. 将系统划分成子系统.....</b>     | <b>73</b> |
| 1. 移动客户端子系统的划分.....            | 73        |
| 1.1. 媒体采集与处理子系统.....           | 73        |
| 1.2. 播放与互动子系统.....             | 73        |
| 1.3. 本地数据管理子系统.....            | 73        |

|                               |    |
|-------------------------------|----|
| 2. 云端服务子系统的划分.....            | 73 |
| 2.1. 用户中心子系统.....             | 74 |
| 2.2. 视频内容管理子系统.....           | 74 |
| 2.3. 社交互动子系统.....             | 74 |
| 2.4. 智能推荐子系统.....             | 74 |
| 2.5. 内容安全与审核子系统.....          | 75 |
| 3. 子系统间的依赖与通信关系.....          | 75 |
| 4. 物理部署映射.....                | 75 |
| 5.5. 确定问题内部的并发性.....          | 75 |
| 1. 移动客户端的并发设计.....            | 75 |
| 2. 服务端的并发设计.....              | 76 |
| 3. 对象状态与互斥管理.....             | 77 |
| 4. 边界条件的并发处理.....             | 77 |
| 5.6. 配置子系统的硬件.....            | 77 |
| 1. 物理节点分类与配置.....             | 78 |
| 1.1. 移动终端节点.....              | 78 |
| 1.2. 应用服务计算节点.....            | 78 |
| 1.3. 媒体处理与智能计算节点.....         | 78 |
| 1.4. 数据存储节点.....              | 79 |
| 1.5. 边缘分发节点.....              | 79 |
| 2. 网络连接与拓扑结构.....             | 79 |
| 3. 硬件冗余与故障转移.....             | 79 |
| 5.7. 管理数据存储.....              | 80 |
| 1. 非结构化数据存储：分布式对象文件系统.....    | 80 |
| 2. 结构化核心数据存储：关系型数据库.....      | 80 |
| 3. 高频互动数据存储：内存数据库.....        | 81 |
| 4. 大数据日志存储：列式数据库.....         | 81 |
| 5. 数据访问层的抽象设计.....            | 81 |
| 5.8. 处理全局资源.....              | 82 |
| 1. 全局唯一标识符（UUID）生成策略.....     | 82 |
| 2. 访问控制与身份认证资源.....           | 82 |
| 3. 网络带宽资源的配额管理.....           | 82 |
| 4. 数据库连接与线程池管理.....           | 83 |
| 5. 全局配置中心.....                | 83 |
| 5.9. 选择软件控制策略.....            | 83 |
| 1. 移动客户端：事件驱动型控制策略.....       | 84 |
| 2. 云端 API 服务：过程驱动与并发控制结合..... | 84 |
| 3. 视频后台处理：管道与过滤器控制策略.....     | 85 |

|                        |           |
|------------------------|-----------|
| 4. 复杂交互逻辑：有限状态机控制..... | 85        |
| 5. 异常与中断处理策略.....      | 86        |
| 5.10. 处理边界条件.....      | 86        |
| 1. 初始化.....            | 86        |
| 2. 终止.....             | 87        |
| 3. 故障与失效.....          | 87        |
| 5.11. 制订复用计划.....      | 88        |
| 1. 框架级复用.....          | 88        |
| 2. 外部类库与组件复用.....      | 89        |
| 3. 内部组件复用与构建.....      | 89        |
| 4. 设计模式复用.....         | 90        |
| <b>第6章 详细设计.....</b>   | <b>91</b> |
| 6.1. 类设计规范与标准.....     | 91        |
| 1. 命名规范.....           | 91        |
| 2. 可见性与访问控制.....       | 91        |
| 3. 数据类型标准.....         | 91        |
| 6.2. 类的详细设计与精化.....    | 91        |
| 1. 实体类精化.....          | 91        |
| 2. 控制类设计.....          | 91        |
| 3. 边界类设计.....          | 92        |
| 6.3. 操作与方法设计.....      | 92        |
| 1. 操作签名定义.....         | 92        |
| 2. 对象生命周期与状态转换.....    | 92        |
| 3. 关键算法逻辑.....         | 92        |
| 6.4. 关联与结构设计.....      | 92        |
| 1. 类之间的关联实现.....       | 92        |
| 2. 依赖管理与解耦.....        | 93        |
| 6.5. 数据存储详细设计.....     | 93        |
| 1. 关系型数据库模式.....       | 93        |
| 2. 非结构化存储设计.....       | 93        |
| 3. 缓存数据结构（Redis）.....  | 93        |
| 6.6. 接口详细说明.....       | 93        |
| 1. 客户端与服务端交互接口.....    | 93        |
| 2. 内部子系统接口.....        | 93        |
| <b>后记.....</b>         | <b>94</b> |
| <b>参考文献.....</b>       | <b>95</b> |

# 第1章 立项

## 1.1. 项目起源与提案

### 1. 发现问题

在当前的移动互联网环境下，我们观察到青少年群体中兴起了一种自发的、富有创造力的内容创作模式：他们使用手机或数码相机录制生活中的精彩片段（如舞蹈、滑板、搞笑短剧等），然后借助另一台设备（如电脑、MP3播放器或另一部手机）播放背景音乐，通过后期剪辑或直接跟拍的方式，将画面与音乐结合，最终将作品上传到论坛、博客或早期视频网站进行分享。这个过程虽然充满热情，但存在明显的技术断层和体验割裂：

**操作繁琐：** 用户需要在不同设备间切换，涉及文件传输、音画对齐等复杂步骤，创作门槛高。

**即时性差：** 无法实现“即想即拍、即拍即享”，灵感与创作冲动在繁琐的流程中被消耗。

**社交孤立：** 分享渠道分散，缺乏一个专注于此类短音乐视频的平台，创作者难以找到同好，无法形成有效的互动和反馈闭环。

### 2. 根源分析

**工具层面：** 缺乏一体化移动端创作工具，音画对齐、特效添加需跨设备操作

**平台层面：** 内容分发依赖主动搜索，无法精准匹配用户兴趣

**生态层面：** 观看与创作场景分离，互动形式浅层，缺乏闭环激励

### 3. 系统边界定义

**内部系统：** 创作工具模块、推荐算法模块、社交互动模块、内容审核模块、运营管理模块

**外部交互：** 音乐版权方接口、云服务供应商、应用商店、电信运营商、监管机构数据上报通道

### 4. 约束确定

**法律约束：** 需遵守音乐版权法规、网络内容合规要求、用户数据隐私保护条例

**技术约束：** 适配 Android 7.0 及以上版本，支持主流机型硬件特性

**资源约束：** 初期开发团队 5 人，首年运营成本控制在 2100 万 - 4800 万

### 5. 提案构想

因此，我们提议开发“乐拍视界”——一款真正实现视频拍摄、智能配乐、一键美化、无缝社交一体化的移动应用程序。

对于视频拍摄者：我们将提供海量正版热门音乐库，用户可以在拍摄前或拍摄后轻松选择配乐；应用内置智能节拍识别功能，能自动将视频画面与音乐高潮点对齐；提供多种电影级的滤镜和转场特效，让零基础的普通用户也能在几分钟内创作出酷炫的、富有感染力的音乐短视频。

对于内容消费者与社交分享者，我们致力于打造一个以音乐为纽带、以视频为载体的沉浸式社交平台。平台将用户从传统的“搜索—观看”单向模式中解放出来，转向“发现—互动—再创作”的闭环体验，构建一个真正“懂你”的音乐视频互动社区：

在内容层面，我们通过强大的个性化推荐算法，实现从“人找内容”到“内容找人”的转变。系统会根据用户的观看偏好、互动行为与音乐口味，智能推送感兴趣的视频，降低选择成本，提升沉浸感。

在社交层面，我们重视人与人之间的连接与互动。用户可通过“同款音乐”功能进行创意比拼，通过点赞、评论、分享和关注等行为，与其他创作者建立联系。形成一个紧密的、充满活力的创意社群。这些互动能够促进内容的流动与传播。

## 1.2. Business Case

### 1. 摘要

该项目瞄准 15-30 岁的年轻群体，旨在解决其制作高质量、富有表现力的音乐短视频时面临的“操作复杂、社交低效”的核心痛点。通过将专业的视频拍摄、庞大的音乐库、智能的剪辑工具与强大的社交功能无缝整合，“乐拍视界”将开创“移动短音乐视频社交”这一全新赛道。我们预期通过快速获取用户、构建内容生态，并最终通过广告、虚拟商品、直播和品牌合作等多种方式实现盈利，占据市场领导地位。

### 2. 市场机遇

移动设备普及：智能手机性能不断提升，网络开始普及，为移动端视频的拍摄、上传和播放提供了硬件基础。

用户行为变迁：年轻一代是“视觉系”原住民，他们更倾向于用图像和视频而非文字进行表达和社交。

音乐需求旺盛：音乐是年轻人表达情绪、彰显个性的重要载体，与视频结合拥有巨大的想象空间。

**市场空白：**现有社交平台（如微博、QQ空间）或视频平台（如优酷、Youtube）均未提供专为“短音乐视频”设计的、便捷的观看、创作与社交一体化体验。

### 3. 目标市场与客户细分

**核心用户：**15-25岁的学生和年轻白领。他们追求潮流、乐于表达、渴望认同、拥有强烈的社交需求。

**次要用户：**25-30岁的都市青年，以及有记录和分享孩子成长瞬间需求的年轻父母。

**潜在用户：**寻求与年轻消费者建立连接的品牌方、广告主，以及希望通过内容创作获得影响力的创作者/KOL。

### 4. 竞争优势

**产品体验优势：**打开即播放，单列信息流，上下滑动切换的模式让观看视频简单轻松。提供低门槛的创作体验，将海量正版音乐库、一键式美颜滤镜、丰富的特效模板整合进App。用户不需要专业技巧，就能轻松创作出看起来“很酷”的视频。一体化产品体验，构建了从发现、音乐选择、特效拍摄、一键发布到互动分享的极致流畅闭环，其无缝体验显著优于功能割裂的传统音视频工具。

**技术与算法优势：**该产品采用基于深度学习的推荐算法，能通过用户极短时间的观看行为（完播率、点赞、评论、转发、停留时长等），精准地建模用户兴趣，并实时调整推荐内容。并且可以给用户在拍摄时提供滤镜、美颜、贴纸、背景、时间特效（如慢动作、快动作）等功能。

### 5. 市场营销与用户获取策略

**冷启动：**从艺术院校、舞蹈社团、街舞爱好者等垂直社群切入，邀请种子用户，生产高质量内容。

**校园推广：**与全国高校合作，举办“校园音乐视频大赛”，快速在目标人群中建立知名度。

**线上营销：**在微博、贴吧、QQ空间等年轻人聚集的社交平台进行内容投放和话题炒作。

### 6. 风险与应对

**竞争风险：**其他软件的模仿与跟进。对策：以快速迭代产品，不断更新技术，提升用户体验为基础，花重金提高产品知名度，拓展商业合作，最终实现市场上的稳固地位。

**版权风险：**音乐版权纠纷。对策：初期与音乐版权代理商建立正规合作，后期建立自己的版权库。

内容风险：用户上传违规内容。对策：建立“AI 算法+人工审核”的内容审核机制，确保内容健康。

盈利风险：无法高效盈利。对策：谨慎探索多元化的商业模式，优先保障用户体验。

## 7. 成本估算

一次性开发成本：

开发团队 5 人，打造一个基础版本约需 6-12 个月，按平均月薪 1 万计算，1 个月约 5 万。初期用户量少，采用云服务（如阿里云、腾讯云）。视频存储和带宽是主要开销，初期每月约 1-3 万。最低开发成本最低为 36 万，最高为 96 万。

持续运营成本：

团队扩充至 30-40 人，新增运营、市场、算法、审核等岗位，年度人力成本约 800 万-1200 万。

带宽与服务器成本（随用户量指数增长）：若日活达到百万级，月度带宽成本可能达到 50 万-200 万，年度 600 万-2400 万。

市场营销费用：用户获取与品牌建设，预估 500 万-1000 万。

音乐版权与内容审核：预估 1 年 200 万。

首年持续运营总成本预估：2100 万-4800 万人民币。

## 8. 项目目标

短期目标（6-12 个月）：成功上线 Android 版本，积累首批核心种子用户，建立活跃的创作者社区，实现每日用户创作视频量过万。

中期目标（1-2 年）：成为国内青少年群体中最受欢迎的短音乐视频社交平台，形成独特的社区文化，探索初步的商业化模式。

长期目标（3-5 年）：构建以“乐拍视界”为核心的短视频内容生态，成为引领年轻文化潮流的重要阵地，并拓展至海外市场。

## 第2章愿景

### 2.1. 问题陈述

#### 1. 问题一

| 要素 | 描述   |
|----|--|
| 问题 | 普通用户的创作过程割裂且技术门槛高，缺乏一个集成化工具，能够让他们在移动端轻松、快速地将视频与热门音乐结合，并添加专业特效的创作。现有流程依赖多个割裂的设备和复杂软件，操作繁琐。  |
| 影响 | 创作者、平台内容生态   |
| 结果 | 用户旺盛的创作热情和灵感被技术难题所压制。繁琐的流程导致用户从“灵感”到“作品”的转化率极低，大量创意被浪费。平台内容生态依赖少数技术娴熟的创作者，内容风格单一，数量增长缓慢。绝大多数潜在创作者保持沉默，无法形成百花齐放的社区氛围。                           |
| 优点 | 该产品提供了一体化移动端创作工具，内嵌海量正版音乐库、智能剪辑功能和丰富特效。<br>实现“所想即所得”，用户只需选择音乐和片段，算法自动完成音画对齐与节奏匹配，将创作门槛降至最低。<br>激发创作欲望，让每个人都能轻松制作出酷炫的音乐短视频，从而极大地丰富了平台内容的多样性和数量。 |

#### 2. 问题二

| 要素 | 描述  |
|----|---|
| 问题 | 内容发现效率低下，用户留存困难。传统平台依赖用户主动搜索和订阅，这是一种“人找内容”的低效模式，无法根据用户的潜在兴趣进行精准内容推荐，导致用户陷入选择疲劳。                 |
| 影响 | 平台运营方，用户  |
| 结果 | 用户难以持续发现感兴趣的内容，浏览体验枯燥。平台无法为用户创造“上瘾”的沉浸感，导致用户使用时长短、流失率高。<br>平台活跃度增长陷入瓶颈，无法让用户稳定留在平台。即使有优质内容，也无法被 |

|    |  |
|----|--|
|    | 对的人看到，内容价值无法最大化。   |
| 优点 | 引入基于 AI 的个性化推荐引擎，通过分析用户行为，精准推送其可能感兴趣的内<br>容，从“人找内容”变为“内容找人”。<br><br>打造全屏沉浸式信息流，提供无缝、零思考的浏览体验，最大化用户的专注度和停<br>留时长。 |

### 3. 问题三

| 要素 | 描述  |
|----|---|
| 问题 | 社交互动薄弱，从观看到创作的转化路径断裂。现有平台的社交互动停留在浅层的点赞和评论，且观看与创作是两个完全割裂的场景。用户即使被视频激发起创作欲望，也缺乏无缝、低成本的创作方式。   |
| 影响 | 用户、平台生态   |
| 结果 | 用户仅是被动的内容消费者，难以深度融入社区。创作灵感因复杂的转化路径而转瞬即逝，平台无法将高涨的观看情绪有效转化为创作行为。<br><br>平台里没有热闹的交流感，更像用户各自刷内容的“冷清空间”，用户之间没形成关联。内容生态缺乏自生长的动力，需要持续的外部刺激来维持内容产出，运营成本高。 |
| 优点 | 构建以“同款音乐”和“挑战赛”为核心的互动机制，用户可一键使用同款模板参与创作，实现“看了就想拍”。<br><br>深度整合社交功能，如好友动态、合拍等，强化用户之间的创意互动与联系。<br><br>形成“观看-灵感-创作-互动”的完美闭环，驱动内容的自我繁荣。               |

## 2.2. 涉众与用户

### 1. 涉众

| 涉众   | 涉众类型 | 简要描述                       |
|------|------|----------------------------|
| 项目经理 | 开发团队 | 制定项目计划，管理进度、风险和资源，保证项目按期交付 |
| 测试人员 | 开发团队 | 测试系统的功能、性能、数据安全            |

|            |        |  |
|------------|--------|--|
| 需求分析师      | 开发团队   | 与用户沟通，获取需求和想法，将这些需求转化为详细的需求规格说明书。                                |
| 编码员        | 开发团队   | 负责编码实现系统   |
| 审核人员       | 平台维护人员 | 审核用户发布的作品，删除违规的作品并提醒作者，处理违规用户                                    |
| 客服         | 平台维护人员 | 解答用户的问题  |
| 官方账号运营团队   | 平台维护人员 | 运营平台的官方账号，通过发布内容、策划线上活动（如挑战赛）来激发社区活力，提升用户粘性                      |
| 国家互联网信息办公室 | 监管机构   | 负责互联网信息内容的管理，落实互联网信息传播的方针政策，指导、协调和督促有关部门加强网络内容管理，并依法查处违法违规网站和应用。 |
| 工业和信息化部    | 监管机构   | 负责电信与互联网行业的管理，监管范围包括网络基础设施、信息服务业务许可、网络与信息安全技术标准等。                |
| 国家版权局      | 监管机构   | 负责监管平台上的版权问题，处理用户上传内容可能涉及的音乐、视频、文字等版权侵权纠纷。                       |
| 公安部        | 监管机构   | 负责网络安全保卫工作，打击利用平台进行的网络诈骗、传播违法信息、侵犯公民个人信息等违法犯罪活动。                 |
| 用户         | 用户     | 观看或发布音视频   |
| 音乐版权方      | 版权方    | 提供音乐的版权，防止音乐作品在未经授权的情况下被平台用户使用                                   |
| 应用商店       | 合作伙伴   | 审核应用资质，确保应用上架符合相关法律法规和平台规定                                       |
| 电信运营商      | 合作伙伴   | 提供网络连接   |
| 云服务商       | 合作伙伴   | 提供云服务  |

|     |      |                                       |
|-----|------|---------------------------------------|
| 品牌方 | 合作伙伴 | 提供钱让平台发布广告，他们是平台未来实现流量变现、进行商业合作的关键对象。 |
| 投资者 | 发起人  | 资金提供者，关注投资回报                          |

## 2. 用户

| 用户   | 用户类型  | 简要描述  |
|------|-------|---|
| 观看者  | 内容消费者 | 寻求轻松、有趣的娱乐方式，打发碎片时间；通过浏览内容获得放松。<br>或为获取知识、技能、资讯，主动搜索或关注科普、技能、资讯类内容，追求内容的专业性、实用性和可学习性。 |
| 官方机构 | 内容创作者 | 宣传活动，发布热点视频吸引流量代表平台运营官方账号，通过策划宣传活动、发布热点视频、输出平台动态等方式吸引流量、传递品牌价值、维护用户关系。                |
| 博主   | 内容创作者 | 渴望进行自我表达，获得关注和认同；需要低门槛、高效的创作工具；希望通过分享日常、生活技巧等内容，与同好交流，积累粉丝。                           |
| 运营团队 | 管理类   | 负责内容生态搭建、创作者扶持与管理，统筹内容审核、流量运营及用户维护，保障平台有序运转与持续增长。                                     |
| 审核人员 | 业务处理类 | 对平台内容、用户行为进行合规性审核，依据平台规则筛查违规信息，保障平台内容生态的健康、安全与合规。                                     |
| 客服   | 业务处理类 | 及时解答用户在平台使用过程中的疑问、投诉与建议，处理用户诉求，提升用户使用体验与满意度。  |

## 2.3. 关键涉众和用户的需要

| 关键涉众   | 需要   |
|--------|--|
| 投资者    | <ol style="list-style-type: none"> <li>商业回报：清晰的盈利路径（如广告、虚拟商品、电商）和投资回报率。</li> <li>增长潜力：高速的用户增长、市场占有率及平台网络效应的形成。</li> <li>竞争壁垒：产品相较于潜在竞争者的独特优势和可持续性（如技术、社区文化）。</li> </ol>      |
| 音乐版权方  | <ol style="list-style-type: none"> <li>版权保护与收益：其音乐作品被合法使用，并能获得公平的版权分成。</li> <li>作品推广：平台能成为其新歌、新艺人推广的有效渠道，扩大音乐影响力。</li> <li>数据洞察：获得其音乐在平台上的使用数据（如播放量、热门视频等），以指导宣发。</li> </ol> |
| 政府监管机构 | <ol style="list-style-type: none"> <li>内容合规：平台内容符合国家安全、社会公序良俗及青少年保护法规。</li> <li>数据安全：用户数据（尤其是未成年人数据）的收集、使用符合相关法律要求。</li> <li>协同治理：平台能积极配合监管要求，建立有效的自查与清理机制。</li> </ol>       |

| 关键用户 | 需要   |
|------|--|
| 观看者  | <ol style="list-style-type: none"> <li>个性化娱乐：平台能“懂我”，通过精准算法持续提供我感兴趣的内容，带来轻松愉悦的“杀时间”体验。</li> <li>社交归属感：能关注喜欢的创作者，与同好互动（点赞、评论），感觉自己身处一个有趣的社区。</li> <li>内容新鲜度：总能发现新的潮流、热门话题和创意形式，保持对平台的新鲜感和期待。</li> </ol> |

|      |   |
|------|---|
|      | <ul style="list-style-type: none"><li>4. 精准获取内容：通过关键词搜索、标签分类快速找到目标内容，过滤无效信息，高效获得想要的内容。</li><li>5. 内容权威性：内容来源可信、逻辑清晰，有实操步骤或权威依据，能真正了解世界，扩充自己。</li><li>6. 视频体系化：支持内容收藏、合集查看，方便按主题连贯观看。</li></ul>  |
| 博主   | <ul style="list-style-type: none"><li>1. 低门槛表达：提供简单易用、功能强大的创作工具（音乐库、特效、模板），让创意能轻松实现。</li><li>2. 获得认可与影响力：获得粉丝、点赞、评论等社交正反馈，积累个人影响力，满足表达欲和成就感。</li><li>3. 创作灵感启发：能方便地追踪热门挑战和流行趋势，获得持续创作的灵感和动力。</li><li>4. 提高粉丝留存度：增加账户热度。</li><li>5. 保证粉丝活跃度：扩大账户知名度。</li></ul> |
| 运营团队 | <ul style="list-style-type: none"><li>1. 生态健康运转：通过数据工具监控内容质量、用户行为，及时调整运营策略，维持正向生态。</li><li>2. 创作者扶持：搭建分层扶持体系，提供流量倾斜、工具特权等资源，激励优质创作者留存。</li><li>3. 增长与转化：策划热点活动、优化流量分发机制，提升用户活跃度、留存率及商业转化效率。</li></ul>   |
| 官方机构 | <ul style="list-style-type: none"><li>1. 高效宣传：借助平台流量优势，让宣传内容精准触达目标受众，提升活动或品牌曝光度。</li><li>2. 权威形象传递：通过官方认证标识、合规内容审核支持，保障信息发布的权威性和可信度。</li><li>3. 互动与反馈：搭建与用户的沟通渠道，收集公众意见，提升品牌好感度和用户粘性。</li></ul>   |
| 审核人员 | <ul style="list-style-type: none"><li>1. 高效审核工具：提供智能筛查、关键词识别等辅助工具，提升违规内容识别效率，降低工作负荷。</li></ul>  |

|    |   |
|----|---|
| 客服 | 2. 明确审核标准：有清晰、统一的规则手册和更新机制，避免审核判断偏差。  |
|    | 3. 风险预警支持：对高风险内容、敏感话题提前预警，保障审核工作的准确性和及时性。   |
|    | <p>1. 高效响应工具：整合用户咨询渠道，提供快捷回复模板、问题分类标签，快速处理用户诉求。</p> <p>2. 问题解决支持：获取平台规则、功能说明等权威资料，能准确解答用户疑问或协调处理复杂问题。</p> <p>3. 用户反馈同步：建立反馈机制，将用户高频问题、建议同步给相关团队，优化服务体验。</p> |

## 2.4. 产品概述

### 1. 产品定位陈述

|             |  |
|-------------|--|
| For         | 渴望表达自我的年轻一代与寻求轻松、愉悦内容消费的移动互联网用户  |
| Who         | 他们需要一种简单、有趣的方式来创作和分享生活，并渴望获得即时的社交互动与认同。但现有工具创作门槛高，平台内容分发效率低下，观看与创作行为割裂。                            |
| The         | 乐拍视界   |
| That        | 是一款集音乐短视频创作、智能推荐与沉浸式社交于一体的移动平台。我们通过一体化创作工具和精准的推荐算法，为用户提供零门槛的创意表达和高度个性化的内容消费体验，让每个人都能轻松记录和分享生活中的乐趣。 |
| Unlike      | 不同于功能复杂、以长视频和搜索为核心的 YouTube，也不同于功能单一、缺乏社交生态的 Dubsmash。   |
| Our product | 我们提供了从创意激发、极简创作到精准分发与互动的完整闭环，构建了一个充满活力的创意社区。   |

### 2. 完整的产品概述

#### 2.1. 能力概述

乐拍视界作为移动端短音乐视频社交应用，核心向用户提供三大核心能力。

创作方面，通过内置海量正版音乐库、智能剪辑工具、美颜滤镜、动态贴纸等功能，以及“一键出片”模板，实现低门槛视频创作与美化；

消费方面，以全屏上下滑动的沉浸式信息流为载体，通过个性化推荐算法，为用户推送定制化内容流；

社交方面，借助点赞、评论、关注、分享、私信等功能。构建以音乐为核心的创作互动与灵感交流闭环，覆盖“创作-消费-社交”全场景需求。

## 2.2. 客户效益 (Benefits)

|          | 涉众类型                | 核心效益                         | 对应产品特性 |
|----------|---------------------|------------------------------|--------|
| 创作者      | 降低创意表达门槛，快速实现创作想法   | 智能音乐匹配、简易拍摄美化、创意特效库、“一键出片”模板 |        |
|          | 获得社交认同与灵感启发，提升创作动力  | “拍同款”功能、同款音乐聚合页              |        |
| 消费者      | 高效获取感兴趣的内容，获得沉浸娱乐体验 | 个性化推荐算法、全屏沉浸式信息流             |        |
|          | 发现潮流内容与同好，增强社区归属感   | 互动功能                         |        |
| 品牌 / 运营方 | 实现品牌宣传与商业收益转化       | 直播带货、广告投放系统                  |        |

## 2.3. 假设和依赖

### 核心假设

1. 用户对短音乐视频的创作需求持续存在，且倾向于“低操作成本、高创意呈现”的创作模式。
2. 个性化推荐算法能精准捕捉用户兴趣，持续提升用户留存与使用时长。
3. 以音乐为核心的社交互动模式，能有效激发用户参与度，形成稳定的社区生态。

### 关键依赖

1. 版权依赖：需持续与主流音乐版权方合作，保障曲库的合法性、丰富度与时效性。
2. 技术依赖：依赖智能节拍识别、推荐算法、实时特效渲染等技术的稳定迭代与优

化。

3. 环境依赖：适配主流移动端操作系统（Android），需兼容不同品牌、型号的手机硬件与系统版本。
4. 生态依赖：需吸引足够数量的创作者产出优质内容，同时积累初始用户群体，形成“创作 - 消费”的正向循环。

## 2.4. 取舍和竞争

### 核心取舍

1. 功能取舍：优先聚焦“音乐 + 短视频”核心场景，简化复杂编辑功能，暂时放弃长视频、多场景综合剪辑等非核心能力，确保创作门槛足够低。
2. 内容取舍：侧重年轻化、潮流化、娱乐化内容生态，暂时弱化专业知识。

### 竞争对手

| 竞争产品    | 优势                  | 劣势                     | 本产品差异化优势                                  |
|---------|---------------------|------------------------|---|
| 快手      | 下沉市场渗透深、社区氛围浓厚、真实感强 | 内容精致度不足、潮流属性弱、推荐精准度待提升 | 信息流更沉浸、创作工具更侧重音乐适配，潮流化内容导向更明确             |
| YouTube | 内容生态丰富、短视频全覆盖、搜索功能强 | 功能复杂、操作门槛高、社交互动性弱      | 聚焦移动端短视频，简化操作流程，强化“创作 - 社交”闭环，更贴合碎片化使用场景。 |

## 2.5. 产品特性

### 1. 内容消费者

| 核心需要              | 对应系统特性                          |
|-------------------|---------------------------------|
| 获得个性化娱乐体验，高效“杀时间” | 精准推荐算法：基于用户兴趣与行为，持续推送感兴趣的短视频。   |
|                   | 全屏沉浸体验：上下滑动无缝切换视频，最大化娱乐沉浸感。     |
| 融入有趣社区，获得社交归属感    | 多元互动工具：提供关注、点赞、评论、私信等功能，构建互动社区。 |

|                    |   |
|--------------------|---|
|                    | 粉丝团体系：支持用户加入专属粉丝团，增强与创作者的联结。  |
| 持续接触新鲜潮流，保持平台新鲜感   | 热点聚合页面：实时更新热门话题与挑战，一站式追踪全网潮流。   |
|                    | 创意模板库：提供海量同款音乐、特效和拍摄模板，降低参与门槛。  |
| 高效获取实用知识或技能，解决实际问题 | 垂直知识库：按用途、技能等分类聚合深度内容。<br>精准搜索筛选：支持关键词搜索，并可按最新、最热等维度筛选。<br>“干货”标识系统：对知识密度高的视频进行标记，便于用户识别。       |
| 视频体系化，避免碎片化        | 合集/列表功能：创作者可将系列内容整理成合集，支持连续观看。<br>视频进度跟踪：自动记录在合集中的观看进度，支持断点续看。<br>笔记与收藏功能：支持在看视频时记录要点，并分类收藏内容。  |
| 辨别信息真伪，获取可靠内容      | 创作者认证体系：对教育、医学等领域的专业人士进行身份认证。<br>事实核查机制：与权威机构合作，对热门技能、知识类视频进行事实标注。<br>优质创作者推荐：在相关领域优先推荐经过认证的用户。 |
| 激发兴趣，探索未知领域        | 知识科普话题：设立如“科普一下”等官方话题，降低认知门槛。<br>跨领域推荐：在用户原有兴趣基础上，智能推荐关联领域的入门内容。                                |

## 2. 内容创作者

| 核心需要               | 对应系统特性   |
|--------------------|--|
| 低门槛实现创意表达，快速完成作品制作 | 海量正版音乐库：提供分类清晰、一键使用的正版音乐与音效。<br>剪辑工具集：添加字幕、添加背景音乐、添加特效、裁剪视频等辅助工具。                            |
| 获得社交正反馈，积累粉丝与影响力   | 实时美颜与滤镜：提供多档美颜调节与风格化滤镜，提升画面质感。<br>“一键出片”模板：提供海量创意模板，替换素材即可快速生成优质视频。                          |
| 获取创作灵感，紧跟行业趋势      | 实时数据看板：清晰展示作品点赞、评论、转发、涨粉等核心数据。<br>粉丝分层管理：支持对粉丝进行分组、标记，实现精细化社群运营。<br>私信互动管理：提供高效的私信收发与批量管理功能。 |
| 提高粉丝留存度与保证粉丝活跃度    | 热门挑战榜单：实时展示平台最热门的挑战与话题。<br>潮流内容推荐：根据创作者领域个性化推送热门内容与同行佳作。<br>同款音乐聚合页：热门 BGM 及使用该音乐的高赞作品集中展示。  |

| 核心需要              | 对应系统特性                           |
|-------------------|----------------------------------|
| 搭建健康内容生态，保障平台合规运转 | 审核任务管理后台：支持任务批量分配、优先级设置与审核员绩效统计。 |

| 核心需要            | 对应系统特性  |
|-----------------|---|
| 扶持优质创作者，提升创作者留存 | 创作数据分析工具：为创作者提供作品、粉丝、收益等多维度数据分析。  |
| 提升用户活跃度与平台增长    | 热点活动策划工具：支持快速创建、发布和推广线上挑战赛等运营活动。<br>用户分层运营模块：基于用户行为标签，实现精准的 Push 与消息触达。 |
|                 | 流量分发调控中心：可对推荐算法策略进行人工干预与权重调整。   |
| 优化商业化转化效率       | 广告投放管理后台：管理广告位库存，监控填充率等核心指标。<br>电商转化数据监测：实时跟踪从内容曝光到商品成交的全链路数据。          |
|                 | 品牌合作管理系统：管理合作项目流程，并评估项目 ROI。  |

#### 4. 业务处理类

##### 4.1. 审核人员

| 核心需要            | 对应系统特性  |
|-----------------|---|
| 高效完成合规审核，降低工作负荷 | 智能预审与分发：系统自动预筛并标记高风险内容，提升审核效率。<br>批量处理功能：支持对同类低风险内容进行一键通过操作和一键拒绝操作。 |
|                 | 审核工作流引擎：根据内容类型和风险等级自动分配任务队列。  |
| 确保审核标准统一，减少判断偏差 | 实时规则查询手册：提供在线、可搜索的详细审核规则与案例库。<br>审核结果抽样复核：系统自动对审核结果进行抽样，由质检         |

|                   |                                 |
|-------------------|---------------------------------|
|                   | 员进行复核。                          |
| 及时预警高风险内容，保障审核准确性 | 敏感话题实时预警：对突发热点事件及相关内容进行自动标记与提权。 |

#### 4.2. 客服

| 核心需要              | 对应系统特性   |
|-------------------|--|
| 快速响应用户诉求，提升问题处理效率 | 智能快捷回复模板：针对常见问题提供标准化回复模板，一键发送。                                       |
| 同步用户反馈，助力产品优化     | 用户反馈标签化收集：支持为每一条用户反馈打上问题类型与优先级标签。<br>反馈数据看板：统计高频问题、用户满意度趋势，并生成周期性报告。 |

#### 2.6. 其他产品需求

| 类别    | 需求描述  |
|-------|---|
| 性能需求  | 1. 响应速度：95%的视频请求应在1秒内开始播放。<br>2. 流畅性：App主界面滑动及视频播放帧率应稳定在60fps。<br>3. 并发能力：系统需支持百万级日活用户的并发访问与内容上传。 |
| 可用性需求 | 1. 直观易用：新用户无需教程即可完成首次视频发布。<br>2. 一致性：全平台保持统一的UI/UX设计语言和交互逻辑。<br>3. 无障碍：支持系统级字体大小调整，关键元素有足够的对比度。   |
| 可靠性需求 | 1. 系统稳定性：App崩溃率低于0.1%。<br>2. 数据持久性：用户数据与创作内容需有99.9%的可靠性保障。  |
| 安全性需求 | 1. 数据安全：用户密码加密存储，通信链路全程加密。<br>2. 内容安全：具备反垃圾、反作弊、反爬虫机制。<br>3. 隐私保护：严格遵循隐私法规，提供隐私设置开关，明确告知数据用途。     |
| 兼容性需求 | 1. 系统版本：支持Android 7.0及以上版本。<br>2. 机型适配：适配市场主流品牌及全面屏、刘海屏等特殊屏幕。                                     |

## 2.7. 特性优先级

| 编号 | 特性                                | 优先级    |
|----|-----------------------------------|--------|
| 1  | 精准推荐算法：基于用户兴趣与行为，持续推送感兴趣的短视频。     | Must   |
| 2  | 全屏沉浸体验：上下滑动无缝切换视频。                | Must   |
| 3  | 多元互动工具：提供关注、点赞、评论、私信等功能           | Must   |
| 4  | 粉丝团体系：支持用户加入专属粉丝团。                | Should |
| 5  | 热点聚合页面：实时展示平台最热门的挑战与话题。           | Should |
| 6  | 创意模板库：提供海量同款音乐、特效和拍摄模板，降低参与门槛。    | Must   |
| 7  | 垂直知识库：按用途、技能等分类聚合深度内容。            | Should |
| 8  | 精准搜索筛选：支持关键词搜索，并可按最新、最热等维度筛选。     | Must   |
| 9  | “干货”标识系统：对知识密度高的视频进行标记，便于用户识别。    | Must   |
| 10 | 合集/列表功能：创作者可将系列内容整理成合集，支持连续观看。    | Must   |
| 11 | 观看进度跟踪：自动记录在合集中的观看进度，支持断点续看。      | Must   |
| 12 | 收藏功能：收藏重要或喜欢的内容，并可分类收藏内容。         | Must   |
| 13 | 笔记功能：支持在看视频时记录重要时间戳，为该视频添加对应文本内容。 | Could  |
| 14 | 创作者认证体系：对教育、医学等领域的专业人士进行身份认证。     | Must   |
| 15 | 事实核查机制：与权威机构合作，对热门知识类视频进行事实标注。    | Won't  |
| 16 | 优质创作者推荐：在相关领域优先推荐经过认证的用户。         | Could  |
| 17 | 提供正版音乐库：提供分类清晰、一键使用的正版音乐与音效。      | Must   |
| 18 | 剪辑工具集：添加字幕、添加背景音乐、添加特效、裁剪视频等辅     | Must   |

|    |                                      |        |
|----|--------------------------------------|--------|
|    | 助工具。                                 |        |
| 19 | 实时美颜与滤镜： 提供多档美颜调节与风格化滤镜， 提升画面质感。     | Must   |
| 20 | “一键出片” 模板： 提供海量创意模板， 替换素材即可快速生成优质视频。 | Could  |
| 21 | 实时数据看板： 清晰展示作品点赞、评论、转发、涨粉等核心数据。      | Must   |
| 22 | 粉丝分层管理： 支持对粉丝进行分组、标记， 实现精细化社群运营。     | Must   |
| 23 | 私信互动管理： 提供高效的私信收发与批量管理功能。            | Must   |
| 24 | 同款音乐聚合页： 热门 BGM 及使用该音乐的高赞作品集中展示。     | Must   |
| 25 | 商品挂载功能： 支持在视频和直播中挂载商品， 直接引导销售。       | Must   |
| 26 | 直播带货工具集： 提供直播间商品橱窗、优惠券、抽奖等营销工具。      | Must   |
| 27 | 品牌合作接单平台： 为创作者与品牌方提供官方、安全的合作对接渠道。    | Must   |
| 28 | 广告分成的接口： 创作者参与流量分成， 从平台广告收入中获益。      | Could  |
| 29 | 智能审核系统： 应用 AI 模型对文本、图像、视频进行违规内容预筛查。  | Could  |
| 30 | 审核任务管理后台： 支持任务批量分配、优先级设置与审核员绩效统计。    | Must   |
| 31 | 创作者成长体系： 设计等级与权益， 对应不同的流量扶持与工具特权。    | Should |
| 32 | 创作数据分析工具： 为创作者提供作品、粉丝、收益等多维度数据分析。    | Must   |
| 33 | 流量分发调控中心： 可对推荐算法策略进行人工干预与权重调整。       | Must   |

|    |                                      |        |
|----|--------------------------------------|--------|
| 34 | 用户分层运营模块： 基于用户行为标签，实现精准的 Push 与消息触达。 | Could  |
| 35 | 广告投放管理后台： 管理广告位库存，监控填充率等核心指标。        | Should |
| 36 | 电商转化数据监测： 实时跟踪从内容曝光到商品成交的全链路数据。      | Should |
| 37 | 品牌合作管理系统： 管理合作项目流程，并评估项目投资回报率。       | Won't  |
| 38 | 批量处理功能： 支持对同类低风险内容进行一键通过操作和一键拒绝操作。   | Should |
| 39 | 审核工作流引擎： 根据内容类型和内容标签自动分配任务队列。        | Should |
| 40 | 实时规则查询手册： 提供在线、可搜索的详细审核规则与案例库。       | Must   |
| 41 | 审核结果抽样复核： 系统自动对审核结果进行抽样，由质检员进行复核。    | Should |
| 42 | 敏感话题实时预警： 对突发热点事件及相关内容进行标记，并上报。      | Could  |
| 43 | 智能快捷回复模板： 针对常见问题提供标准化回复模板，一键发送。      | Must   |
| 44 | 用户反馈标签化收集： 支持为每一条用户反馈打上问题类型与优先级标签。   | Should |
| 45 | 反馈数据看板： 统计高频问题、用户满意度趋势，并生成周期性报告。     | Should |

## 2.8. 补充说明

### 1. 法律 / 规定性需求

音乐版权：所有配乐需获得版权方合法授权，建立版权分成机制

内容合规：符合《网络安全法》《未成年人保护法》《互联网信息服务管理办法》

数据隐私：遵循个人信息保护相关法规，明确告知用户数据收集与使用用途

### 2. 应用开发标准

开发遵循 Rational Unified Process (RUP) 规范

代码开发符合 Android 应用开发最佳实践

文档编写遵循统一模板，确保可读性与一致性

### 3. 质量属性需求

可用性：系统全年可用性 $\geq 99.9\%$ ，故障恢复时间 $\leq 1$  小时

可靠性：单用户日均崩溃次数 $\leq 0.01$  次

性能：视频上传速度 $\geq 1\text{MB/s}$  (4G 网络环境)，页面加载时间 $\leq 2$  秒

可支持性：提供完整的技术文档，支持远程故障排查

### 4. 设计与实现约束

技术栈：前端采用 QML 开发，后端采用 C++ 开发

部署环境：依赖阿里云服务器与存储服务

兼容性：适配市场占有率前 20 的 Android 机型，支持全面屏、刘海屏等特殊屏幕

### 5. 其他补充需求

闲置处理：用户会话闲置超过 20 秒时，系统自动降低视频清晰度以节省流量；闲置超过 5 分钟，弹出互动提示

未成年人保护：提供青少年模式，过滤不适宜内容，限制使用时长

第3章 用况建模

### 3.1. 术语表

### 3.2. 参与者清单

#### 1. 识别主参与者

| 用户类型        | 参与者      |
|-------------|----------|
| 内容消费者、内容创作者 | 普通用户     |
| 业务处理者       | 内容审核员、客服 |
| 管理类         | 运营人员     |

#### 2. 参与者简要描述

##### 2.1. 普通用户

普通用户是系统的核心使用者，可以在“乐拍视界”中浏览推荐视频流，并在浏览过程中进行创作、社交互动、私信交流等操作。

##### 2.2. 内容审核员

内容审核员负责对用户上传的音乐短视频、评论及互动内容进行合规性审查，依据平台规范对违规内容进行处理，以确保平台内容生态的健康合规与安全。

##### 2.3. 客服

客服负责通过系统接收和处理用户反馈、投诉、咨询等各类问题，与用户沟通并提供解决方案，提升用户体验和满意度。

##### 2.4. 运营人员

运营人员负责平台内容生态建设、用户运营和活动策划，通过运营手段提升用户活跃度、留存率和平台影响力。



图 3-1 “乐拍视界” 中的参与者

### 3.3. 乐拍视界的普通用户主要用况

普通用户只有一个主用况：浏览个性化视频流。该用况覆盖用户从启动应用到浏览内容的全过程，包括可能的可选操作（如创作、社交等）。创作音乐短视频、参与社交活动、私信、管理个人资料和内容作为这个用况的备选流。

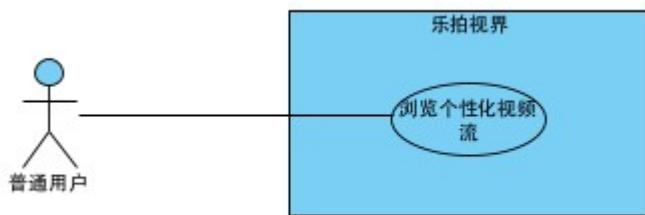


图 3-2 “乐拍视界” 中的普通用户主要用况

### 1. 浏览个性化视频流用况

#### 1.1. 简要描述

普通用户浏览系统根据其兴趣偏好推荐的个性化视频流，通过上下滑动切换视频，获得娱乐和信息消费体验。在浏览过程中，用户可选择进行创作、社交互动、私信或管理个人资料等可选操作。

#### 1.2. 前置条件

1. 用户已成功登录系统
2. 系统有足够的视频内容可供推荐

#### 1.3. 后置条件

1. 用户的互动行为被正确记录
2. 推荐算法根据用户行为更新兴趣模型
3. 系统记录用户的浏览历史

#### 1.4. 基本事件流

{启动应用}

1. 用况开始于用户打开应用主界面

{生成推荐列表}

2. 系统根据用户兴趣偏好生成个性化视频推荐列表

{显示视频}

3. 系统显示第一个推荐视频，开始自动播放

{切换视频}

4. 用户上下滑动屏幕切换视频

{更新用户画像}

5. 系统记录用户的浏览行为（如观看时长、滑动行为）并更新兴趣模型

{退出应用}

6. 用户退出应用时，用况结束

1.5. 备选流

#### A1 处理网络连接问题

在 {启动应用} 或 {生成推荐列表} 时，如果设备无网络连接：

1. 系统显示离线提示信息
2. 系统提供缓存的推荐内容供用户浏览
3. 用况继续执行基本流

#### A2 处理新用户或无历史数据

在 {生成推荐列表} 时，如果用户是新用户或缺乏历史行为数据：

1. 系统基于热门内容、通用偏好生成默认推荐
2. 用况继续执行基本流

#### A3 处理视频加载失败

在 {显示视频} 或 {切换视频} 时，如果视频内容加载失败：

1. 系统显示加载失败提示
2. 系统自动跳过该视频并显示下一个推荐视频
3. 用况继续执行基本流

#### A4 处理内容举报

在 {显示视频} 时，如果用户选择举报不当内容：

1. 系统记录举报信息
2. 系统将举报内容转交审核队列
3. 系统显示举报成功确认
4. 用况继续执行基本流

#### A5 用户选择创作音乐短视频

在{显示视频}时，如果用户选择创作视频：

1. 系统显示创作模式选择界面（拍摄或上传）
2. 用户选择创作模式并授予必要权限
3. 系统提供相应的创作工具
4. 用户录制或选择视频素材
5. 系统提供音乐选择界面
6. 用户浏览音乐库并选择背景音乐
7. 系统自动将视频与音乐节拍对齐
8. 用户使用编辑工具添加特效、滤镜、文字等
9. 用户设置视频标题、描述和可见性
10. 系统验证视频内容符合基本规范
11. 用户确认发布视频
12. 系统将视频提交到内容审核队列
13. 用况继续执行基本流

## A6 用户选择参与社交互动

在{显示视频}时，如果用户选择进行社交互动：

1. 用户选择互动类型（点赞、评论、关注、分享等）和互动对象
2. 系统验证操作权限和内容状态
3. 系统记录互动行为
4. 系统向相关用户发送通知
5. 系统更新社交关系数据
6. 用况继续执行基本流

## A7 用户选择发送私信

在{显示视频}时，如果用户选择发送私信：

1. 用户选择目标联系人
2. 用户编写消息内容

3. 用户发送消息
4. 系统验证消息内容和接收方状态
5. 系统存储消息并标记发送时间
6. 系统向接收方推送新消息通知
7. 系统更新对话列表
8. 用况继续执行基本流

#### A8 用户选择管理个人资料和内容

在{显示视频}时，如果用户选择管理个人资料：

1. 用户进入个人中心界面
2. 用户选择要管理的项目类型（个人信息、收藏内容、发布历史等）
3. 系统显示对应的管理界面和内容
4. 用户执行具体的管理操作
5. 系统验证操作权限和数据完整性
6. 系统更新相关数据和状态
7. 用况继续执行基本流

#### A9 用户搜索视频内容

在 {显示视频} 或 {切换视频} 时，如果用户选择搜索视频：

1. 用户进入搜索界面，输入搜索关键词
2. 系统根据关键词匹配视频标题、描述、标签和创作者信息
3. 系统展示搜索结果列表，按相关性排序
4. 用户选择并观看搜索结果中的视频
5. 系统记录搜索行为和结果点击
6. 用况继续执行基本流

#### 3.4. 乐拍视界的内容审核员主要用况

1. 审核用户发布内容：内容审核员的核心工作，处理系统自动筛选出的疑似违规内容

2. 管理内容审核规则：维护和优化审核系统的工作，确保了审核工作的长期效力和准确性

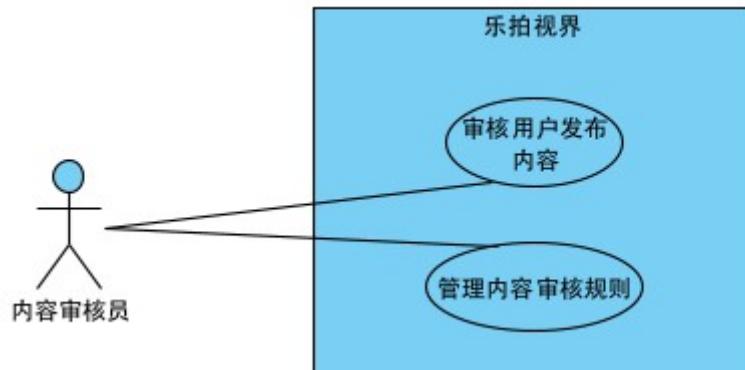


图 3-3 “乐拍视界” 中的内容审核员主要用况

### 1. 审核用户发布内容用况

#### 1.1. 简要描述

内容审核员对系统自动筛查出的疑似违规内容（包括视频、评论、私信等）进行人工复核，并根据平台规范作出相应处理决定。

#### 1.2. 前置条件

1. 内容审核员已成功登录系统
2. 系统内存在待处理（或重新分配）的疑似违规内容队列

#### 1.3. 后置条件

1. 内容项被标记为最终状态（如“通过”、“驳回”）
2. 生成审核操作记录，包括审核员、时间、处理原因等
3. 若内容被驳回，系统会执行相应的处置动作（如删除内容、对发布者发出警告等）

#### 1.4. 基本事件流

{进入审核界面}

1. 用况开始于内容审核员进入审核工作主界面

{分配审核内容}

2. 系统从“待审核队列”中分配（或审核员主动领取）一项待审核内容，并展示内容详情（如视频、文字、发布者信息等）

{审查内容详情}

3. 内容审核员根据平台审核规范，仔细审查内容的各个方面

{做出审核决定}

4. 审核员根据判断做出处理决定

{记录审核结果}

5. 系统记录审核结果，并更新该内容的状态

{结束审核任务}

6. 审核员继续处理下一项内容，或用况结束

#### 1.5. 备选流

##### **A1 处理轻微违规内容**

在 {做出审核决定} 时，如果审核员认定内容属于轻微违规（如用词不雅）：

1. 系统提示审核员选择预设的违规类型
2. 审核员确认违规类型和处理方式
3. 系统除记录结果外，自动向内容发布者发送违规提醒
4. 用况继续执行基本流

##### **A2 标记并移交复杂内容**

在 {做出审核决定} 时，如果审核员认为内容复杂，无法立即做出判断（如可能涉及新型违规手法或法律边界模糊）：

1. 审核员将该内容标记为“待讨论”
2. 系统将内容转交至专家仲裁层或更资深的审核员进行处理
3. 系统记录移交状态和备注信息
4. 用况继续执行基本流

##### **A3 批量处理同类内容**

在 {分配审核内容} 时，如果待审核内容来自同一用户或为高度相似的内容：

1. 审核员选择启用“批量处理”模式
2. 系统展示内容列表和批量操作选项

3. 审核员进行一次判断后，将结果应用于同批所有内容
4. 系统批量更新所有相关内容状态
5. 用况继续执行基本流

#### A4 处理审核标准查询

在 {审查内容详情} 时，如果审核员对审核标准有疑问：

1. 审核员查询相关审核规则和案例
2. 系统显示详细的规则说明和类似案例
3. 审核员基于规则说明做出判断
4. 用况继续执行基本流

2. 管理内容审核规则用况

##### 2.1. 简要描述

具有更高权限的内容审核员（或规则管理员）根据最新政策、业务变化或审核反馈，对系统自动审核所依赖的规则库（如敏感词库、语义模型）进行更新、测试和优化。

##### 2.2. 前置条件

1. 内容审核员已成功登录系统
2. 该审核员拥有"规则管理"的特殊权限

##### 2.3. 后置条件

1. 审核规则库被成功更新
2. 记录规则变更日志（版本、修改人、时间、原因）

##### 2.4. 基本事件流

{进入规则管理}

1. 用况开始于规则管理员进入"审核规则管理"界面

{展示规则列表}

2. 系统展示当前的规则列表、分类及状态

{执行规则操作}

3. 规则管理员执行管理操作，如新增、修改、禁用或启用某条规则

{验证规则变更}

4. 系统验证规则修改的权限和有效性

{保存规则变更}

5. 系统保存更新后的规则，并记录版本日志

{结束规则管理}

6. 用况结束

## 2.5. 备选流

### A1 测试与验证规则效果

在 {执行规则操作} 时，如果规则管理员需要测试新规则效果：

1. 规则管理员选择在测试环境中运行新规则
2. 系统在测试环境中执行规则测试
3. 系统展示新旧规则的对比效果（准确率、效率等）
4. 规则管理员基于测试结果决定是否正式发布规则
5. 用况继续执行基本流

### A2 基于反馈优化规则

在 {执行规则操作} 时，如果规则管理员需要基于实际案例优化规则：

1. 规则管理员查看某条规则的“误判案例”或“漏判案例”统计
2. 系统根据案例分析推荐规则调整方案
3. 规则管理员采纳建议并对规则进行优化
4. 用况继续执行基本流

### A3 导入外部规则库

在 {执行规则操作} 时，如果规则管理员需要批量导入规则：

1. 规则管理员选择导入外部规则文件
2. 系统验证文件格式和内容有效性
3. 系统解析并展示导入的规则内容
4. 规则管理员确认导入内容和冲突解决方案

5. 系统批量更新规则库

6. 用况继续执行基本流

### 3.5. 乐拍视界的客服主要用况

1. 处理用户咨询与投诉：这是客服的核心工作，直接面向用户解决问题

2. 管理客服知识库：这是支撑客服工作的重要后台管理功能

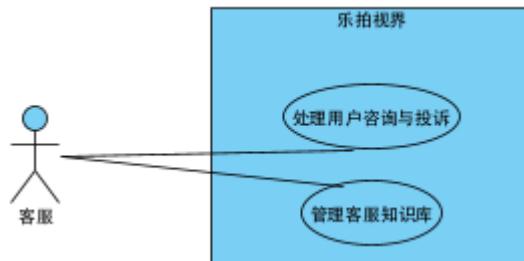


图 3-4 “乐拍视界” 中的客服主要用况

#### 1. 处理用户咨询与投诉用况

##### 1.1. 简要描述

客服通过系统接收用户提交的咨询、投诉、反馈等工单，进行调查分析并与用户沟通，最终解决问题。

##### 1.2. 前置条件

1. 客服已成功登录客服工作台系统

2. 系统内存在待处理的用户工单队列

##### 1.3. 后置条件

1. 用户工单被标记为最终处理状态

2. 生成完整的工单处理记录，包括沟通记录、解决方案等

3. 用户满意度得到记录

##### 1.4. 基本事件流

{进入工作台}

1. 用况开始于客服登录系统并进入客服工作台主界面

{分配工单}

2. 系统从“待处理工单队列”中分配一个新的用户工单给客服

{了解问题}

3. 客服仔细阅读工单内容，了解用户的问题或投诉详情

{联系用户}

4. 客服通过系统提供的沟通渠道与用户进行联系和沟通

{分析解决方案}

5. 客服分析问题原因，调查相关背景信息，并形成解决方案

{执行处理}

6. 客服向用户解释问题原因并提供解决方案，执行必要的处理操作

{关闭工单}

7. 系统记录处理结果，客服关闭工单并记录处理摘要

{结束处理}

8. 用况结束

## 1.5. 备选流

### A1 处理常见问题快速回复

在 {分析解决方案} 时，如果客服识别该问题属于常见问题：

1. 客服从知识库中调用预设的标准回复模板
2. 系统提供一键插入模板内容的功能
3. 客服根据具体情况微调模板内容后发送给用户
4. 用况继续执行基本流

### A2 升级复杂问题至专家支持

在 {分析解决方案} 时，如果客服无法独立解决复杂技术问题：

1. 客服将工单标记为"需要专家支持"
2. 系统将工单转派至专业技术支持团队
3. 系统记录转派原因和当前处理进展
4. 用况继续执行基本流

### A3 处理用户追加反馈

在 {关闭工单} 时，如果用户在工单关闭前追加新的反馈：

1. 系统提示客服有新的用户消息
2. 客服重新与用户沟通，了解追加的问题
3. 根据新情况补充处理方案
4. 用况继续执行基本流

#### A4 处理紧急投诉事件

在 {了解问题} 时，如果客服识别该工单属于紧急投诉：

1. 系统自动提升工单优先级
2. 客服按照紧急投诉处理流程优先处理
3. 系统记录紧急处理过程和结果
4. 用况继续执行基本流

2. 管理客服知识库用况

##### 2.1. 简要描述

客服人员维护和更新客服知识库，包括添加新的常见问题解答、更新解决方案、优化回复模板等，以提升客服团队的工作效率和服务质量。

##### 2.2. 前置条件

1. 客服已成功登录系统
2. 该客服具有知识库管理权限

##### 2.3. 后置条件

1. 知识库内容得到更新和完善
2. 记录知识库变更历史
3. 相关客服人员接收到知识库更新通知

##### 2.4. 基本事件流

{进入知识库管理}

1. 用况开始于客服进入知识库管理界面

{展示知识库}

2. 系统展示知识库分类结构和内容列表  
{执行知识库操作}
3. 客服执行知识库管理操作，如新增、修改或删除知识条目  
{验证操作内容}
4. 系统验证操作权限和内容规范性  
{保存知识库变更}
5. 系统保存知识库变更，更新版本信息  
{结束知识库管理}
6. 用况结束

#### 2.5. 备选流

##### **A1 基于工单统计优化知识库**

在 {执行知识库操作} 时，如果客服需要基于实际工单数据优化知识库：

1. 系统提供工单统计和分析数据
2. 客服识别高频问题和解决方案缺口
3. 基于分析结果新增或修改知识库条目
4. 用况继续执行基本流

##### **A2 处理知识库内容审核**

在 {验证操作内容} 时，如果系统检测到知识库内容需要审核：

1. 系统将新增或修改的内容标记为"待审核"
2. 知识库管理员或资深客服进行内容审核
3. 审核通过后内容正式发布到知识库
4. 用况继续执行基本流

##### **A3 批量导入知识库内容**

在 {执行知识库操作} 时，如果客服需要批量更新知识库：

1. 客服选择批量导入功能
2. 系统提供导入模板和格式要求

3. 客服上传整理好的知识库文件
4. 系统解析并验证导入内容
5. 系统批量更新知识库内容
6. 用况继续执行基本流

#### A4 设置知识库权限和可见性

在 {执行知识库操作} 时，如果客服需要设置不同内容的访问权限：

1. 客服设置知识条目的可见性范围（如全员可见、仅客服可见等）
2. 系统根据设置更新权限控制
3. 不同角色的用户只能看到相应权限的内容
4. 用况继续执行基本流

### 3.6. 乐拍视界的运营人员主要用况

1. 策划并执行运营活动：通过策划挑战赛、话题活动等提升用户参与度
2. 管理内容推荐策略：优化内容分发和推荐机制
3. 监控平台运营数据：通过数据分析指导运营决策

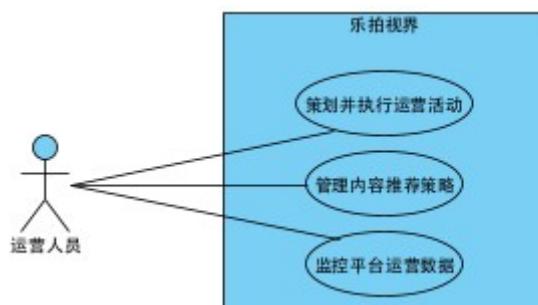


图 3-5 “乐拍视界” 中的运营人员主要用况

#### 1. 策划并执行运营活动用况

##### 1.1. 简要描述

运营人员策划、创建、发布和监控线上运营活动（如音乐挑战赛、话题活动等），旨在提升用户参与度和内容产出。

##### 1.2. 前置条件

1. 运营人员已成功登录运营管理后台
2. 运营人员具有活动策划和发布权限

### 1.3. 后置条件

1. 运营活动成功创建并发布到平台
2. 活动数据监控机制已建立
3. 活动效果报告生成

### 1.4. 基本事件流

{进入活动管理}

1. 用况开始于运营人员进入活动管理界面

{创建活动}

2. 运营人员创建新的运营活动，填写活动基本信息（名称、主题、时间等）

{验证活动信息}

3. 系统验证活动信息的完整性和合理性

{配置活动规则}

4. 运营人员配置活动规则、奖励机制和展示位置

{设置监控指标}

5. 运营人员设置活动数据监控指标和报告频率

{发布活动}

6. 运营人员发布活动，系统将活动推送到指定用户群体

{监控活动效果}

7. 系统开始收集活动参与数据，生成实时监控报告

{结束活动管理}

8. 用况结束

### 1.5. 备选流

#### A1 调整活动策略

在 {监控活动效果} 时，如果活动参与度未达预期：

1. 运营人员分析活动数据，识别问题原因
2. 运营人员调整活动规则或奖励机制

3. 系统应用调整并重新推送活动信息

4. 用况继续执行基本流

## A2 处理活动异常情况

在 {监控活动效果} 时，如果发现异常参与行为（如刷量、违规内容）：

1. 系统自动标记异常参与行为
2. 运营人员审查异常情况并采取处理措施
3. 系统清理异常数据并更新活动统计
4. 用况继续执行基本流

## A3 延长或提前结束活动

在 {监控活动效果} 时，如果需要调整活动时间：

1. 运营人员根据活动效果决定延长或提前结束活动
2. 系统更新活动时间设置并通知参与用户
3. 用况继续执行基本流

2. 管理内容推荐策略用况

### 2.1. 简要描述

运营人员通过调整推荐算法参数、设置内容权重、管理热门话题等方式，优化平台内容分发效果和用户体验。

### 2.2. 前置条件

1. 运营人员已成功登录运营管理后台
2. 运营人员具有推荐策略管理权限

### 2.3. 后置条件

1. 推荐策略配置得到更新
2. 系统按照新的策略进行内容分发
3. 记录策略变更历史

### 2.4. 基本事件流

{进入策略管理}

1. 用况开始于运营人员进入推荐策略管理界面  
{展示当前策略}
2. 系统展示当前推荐算法的配置参数和效果指标  
{分析推荐效果}
3. 运营人员分析推荐效果数据，识别优化机会  
{调整策略参数}
4. 运营人员调整推荐算法参数或内容权重设置  
{验证参数变更}
5. 系统验证参数变更的合理性和安全性  
{应用新策略}
6. 系统应用新的推荐策略，在部分用户群体中进行 A/B 测试  
{监控策略效果}
7. 系统监控新策略的效果数据，与旧策略进行对比分析  
{结束策略管理}
8. 用况结束

## 2.5. 备选流

### A1 处理内容多样性问题

在 {分析推荐效果} 时，如果发现推荐内容同质化严重：

1. 运营人员调整多样性参数，增加内容探索权重
2. 系统重新计算推荐结果，增加新类型内容的曝光
3. 用况继续执行基本流

### A2 紧急干预热点内容

在 {调整策略参数} 时，如果需要紧急推广或抑制特定内容：

1. 运营人员设置特定内容的权重系数
2. 系统立即调整该内容在推荐流中的位置
3. 用况继续执行基本流

## A3 回滚策略变更

在 {监控策略效果} 时，如果新策略导致关键指标下降：

1. 运营人员决定回滚到之前的策略版本
2. 系统恢复之前的参数配置
3. 用况继续执行基本流

3. 监控平台运营数据用况

### 3.1. 简要描述

运营人员通过数据看板监控平台关键运营指标，分析用户行为趋势，发现运营问题并制定相应优化策略。

### 3.2. 前置条件

1. 运营人员已成功登录运营管理后台
2. 系统数据收集和处理功能正常运行

### 3.3. 后置条件

1. 运营人员获得平台运营状况的全面了解
2. 识别出需要优化的运营问题
3. 制定相应的优化行动计划

### 3.4. 基本事件流

{进入数据看板}

1. 用况开始于运营人员进入运营数据看板界面

{展示核心指标}

2. 系统展示关键运营指标（DAU、留存率、内容产出等）的实时数据

{分析数据趋势}

3. 运营人员分析指标趋势，识别异常波动或潜在问题

{深入分析问题}

4. 运营人员钻取详细数据，深入分析问题原因

{制定优化方案}

5. 运营人员基于分析结果制定优化建议或行动计划

{记录分析结果}

6. 系统记录分析结果和优化方案

{结束数据分析}

7. 用况结束

### 3.5. 备选流

#### A1 处理数据异常告警

在 {展示核心指标} 时，如果系统检测到关键指标异常：

1. 系统触发异常告警，突出显示问题指标
2. 运营人员立即查看告警详情，分析异常原因
3. 运营人员制定紧急应对措施
4. 用况继续执行基本流

#### A2 生成周期性运营报告

在 {记录分析结果} 时，如果需要生成正式运营报告：

1. 运营人员选择报告模板和时间范围
2. 系统自动生成包含图表和分析的运营报告
3. 运营人员审核并完善报告内容
4. 系统分发报告给相关利益相关者
5. 用况继续执行基本流

#### A3 设置数据监控告警

在 {分析数据趋势} 时，如果发现需要持续监控的关键模式：

1. 运营人员设置自定义数据监控规则和告警阈值
2. 系统保存监控规则，开始持续监控
3. 当触发告警条件时系统自动通知运营人员
4. 用况继续执行基本流

## 第4章需求分析

### 4.1. 分析模型概述

旨在生成“乐拍视界”短视频社交平台的分析模型。该模型核心目的是在忽略物理实现细节（如具体编程语言、数据库类型或特定 UI 控件）的前提下，对系统的逻辑结构和业务行为进行详细说明。

#### 1. 分析模型的目标与关注点

分析模型关注的是“乐拍视界”系统的逻辑行为。它通过分析问题情景的逻辑结构以及各个逻辑元素相互交互的方式，详细说明各项需求。

1. 逻辑视角：本模型将从整体上建模“乐拍视界”所期望的应用系统业务行为，建立独立于任何特定实现方法的逻辑结构。
2. 完善需求：通过分析，我们将检查不同需求之间可能存在的冲突，确保对“乐拍视界”涉及的人（如普通用户、审核员）、事（如视频发布、审核流程）和概念（如推荐算法策略）有清晰、无歧义的理解。
3. 设计基础：本模型将旨在减少软件设计和构建中的错误与不一致性。

#### 2. 分析模型的组成

本分析模型主要由一组分析类组成，这些类通过协作来实现定义的用况。我们将采用健壮性分析（Robustness Analysis）技术，将类划分为以下三种标准衍型（Stereo-types）：

1. 边界类（Boundary Classes）：用于建模“乐拍视界”系统与其参与者（如观看者、创作者、运营人员）之间的交互。它们代表了系统与外部世界的逻辑接口（例如：视频浏览界面、审核后台界面），主要负责管理跨越系统边界的信息传送。
2. 实体类（Entity Classes）：用于建模“乐拍视界”应用领域中的核心现象或概念的信息及相关行为。这些类通常源自领域模型，代表了需要系统永久存储的信息（例如：短视频、音乐信息、用户信息、审核记录）。
3. 控制类（Control Classes）：用于表示用况逻辑中的协调、排序、事务及控制。它们充当边界类与实体类之间的“胶水”，负责处理特定于用况的计算和调度方面（例如：处理推荐流的分发逻辑、协调视频上传与转码流程）。

#### 3. 用况的实现（Realization）

通过用况实现来构建模型。对于“乐拍视界”的关键用况（如“浏览个性化视频流”、“审核用户发布内容”等），我们将：

1. 识别参与该用况的一组分析类（协作）；
2. 使用通信图（Communication Diagram）来展示这些对象如何在特定情景下通过消息传递进行动态交互；
3. 最终整合形成完整的分析类图，展示类之间的静态结构、关联及多重性。

## 4.2. 健壮性分析

根据 Jacobson 的分类方法，我们将分析模型中的类分为三种特定的行型：边界类（Boundary）、实体类（Entity）和控制类（Control）。

### 1. 边界类

边界类用于对系统与外部世界（参与者）之间的交互进行建模。在“乐拍视界”中，边界类并不直接表示具体的 UI 控件（如按钮或窗口），而是代表系统主要的逻辑接口，负责处理输入和输出。

根据参与者和用况，识别出以下关键边界类：

#### 1. MainFeedUI（主页视频流界面）

关联用况：浏览个性化视频流

描述：负责向普通用户展示全屏沉浸式视频流，捕获用户的滑动（切换视频）、点赞、评论等交互操作，并将请求传递给系统。

#### 2. VideoCreationUI（创作工具界面）

关联用况：浏览个性化视频流（备选流：用户选择创作音乐短视频）

描述：提供拍摄、音乐选择、特效编辑的交互入口，负责接收用户录制的视频数据和编辑指令。

#### 3. AuditWorkbenchUI（审核工作台界面）

关联用况：审核用户发布内容

描述：面向内容审核员的界面，负责展示待审核的视频内容及违规详情，并接收审核员提交的“通过”或“驳回”决定。

#### 4. OperationsDashboardUI（运营管理后台界面）

关联用况：策划并执行运营活动、监控平台运营数据

描述：面向运营人员的界面，用于输入活动规则、发布挑战赛以及展示关键运营指标的

数据看板。

## 2. 实体类

实体类用于对应用领域中的现象、概念、人或事物的相关信息及行为进行建模。这些类表示系统必须永久存储的信息。在“乐拍视界”中，这些类源自对业务领域的理解。

根据项目文档中的业务描述，识别出以下关键实体类：

### 1. User (用户)

描述：表示平台的注册用户（包括内容消费者和创作者）。系统需存储其基本信息、兴趣画像及社交关系（关注/粉丝）。

### 2. Video (短视频)

描述：核心业务对象。存储视频文件的元数据（如 URL、时长）、统计数据（点赞数、播放量）以及状态（审核中、已发布、已下架）。

### 3. Music (音乐)

描述：表示曲库中的音乐条目。存储版权信息、音频文件链接及对应的节拍点数据，是“同款音乐”功能的基础。

### 4. AuditTask (审核任务)

描述：表示一条待处理的审核工作项。记录送审内容、审核状态、处理人及处理时间，用于追踪审核流程。

### 5. OperationActivity (运营活动)

描述：表示平台发起的挑战赛或话题活动。存储活动规则、时间范围及关联的视频集合。

## 3. 控制类

控制类用于表示协调、排序、事务处理以及对其他对象的控制。它们充当边界对象和实体对象之间的“胶水”，负责将用况中的逻辑行为（特定于用况的计算和调度）与具体的实体数据分离开来。

按照“每个用况通常对应一个控制类”的原则，识别出以下关键控制类：

### 1. FeedController (推荐流控制器)

关联用况：浏览个性化视频流

职责：协调视频流的加载逻辑。它接收来自 MainFeedUI 的请求，调用推荐算法（逻辑），查询 Video 和 User 实体，并返回排序后的视频列表。

## 2. CreationController (创作发布控制器)

关联用况：浏览个性化视频流（备选流：创作）

职责：管理视频制作流程。协调 Music 实体的加载，处理视频合成逻辑，并创建新的 Video 实体，将其状态设置为“待审核”。

## 3. AuditController (审核控制器)

关联用况：审核用户发布内容

职责：管理审核工作流。负责从队列中提取 AuditTask，将数据分发给 AuditWorkbenchUI，并根据审核员的决策更新 Video 实体的状态（如设为“公开”或“违规删除”）。

## 4. ActivityController (活动运营控制器)

关联用况：策划并执行运营活动

职责：处理运营活动的生命周期。负责验证活动规则，创建 OperationActivity 实体，并协调活动内容的推送。

### 4.3. 交互建模

通过建立协作（Collaboration），并使用通信图来展示对象之间的动态交互和消息传递方案。交互设计遵循健壮性分析规则：参与者与边界对象交互，边界对象与控制对象交互，控制对象协调实体对象。

#### 1. 浏览个性化视频

##### 1.1. 参与对象（协作角色）

1. 普通用户 (Actor)：发起浏览请求。

2. :MainFeedUI (Boundary)：负责展示视频流界面，接收用户操作。

3. :FeedController (Control)：负责协调推荐逻辑，获取数据。

4. :RecommendationModel (Entity): 封装推荐算法逻辑，提供视频 ID 列表。
5. :Video (Entity): 存储视频的具体内容 (URL、标题)。
6. :User (Entity): 存储视频创作者的信息 (头像、昵称)。

## 1.2. 通讯图

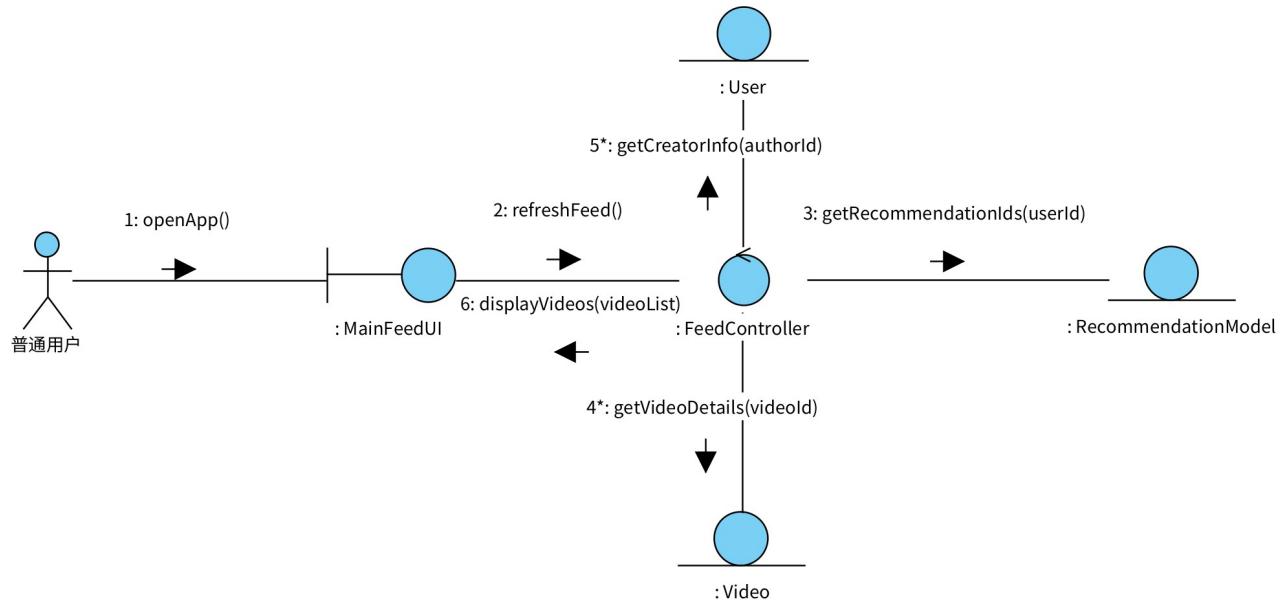


图 4-1 “乐拍视界”中的浏览个性化视频通讯图

### 1. 启动与请求：

1: openApp(): 普通用户启动应用，:MainFeedUI 初始化。

2: refreshFeed(): :MainFeedUI 向 :FeedController 发送刷新请求。

### 2. 获取推荐列表：

3: getRecommendationIds(userId): :FeedController 调用 :RecommendationModel，获取一组推荐的视频 ID 列表。

### 3. 构建视频数据（迭代）：

4\*: getVideoDetails(videoId): :FeedController 根据 ID 列表，向 :Video 对象发送消息，获取视频元数据（如播放地址）。

5\*: getCreatorInfo(authorId): :FeedController 根据视频关联的作者 ID，向 :User (创作者) 对象发送消息，获取作者昵称和头像。

### 4. 展示结果：

6: displayVideos(videoList): :FeedController 将组装好的数据返回给 :Main-

FeedUI，界面渲染视频流供用户观看。

## 2. 审核视频

### 2.1. 参与对象（协作角色）

1. 内容审核员 (Actor): 执行审核操作。
2. :AuditWorkbenchUI (Boundary): 审核工作台界面，展示视频和操作按钮。
3. :AuditController (Control): 管理审核流程状态流转。
4. :AuditTask (Entity): 记录审核任务的状态和分配情况。
5. :Video (Entity): 被审核的视频对象，包含状态属性。

### 2.2. 通讯图

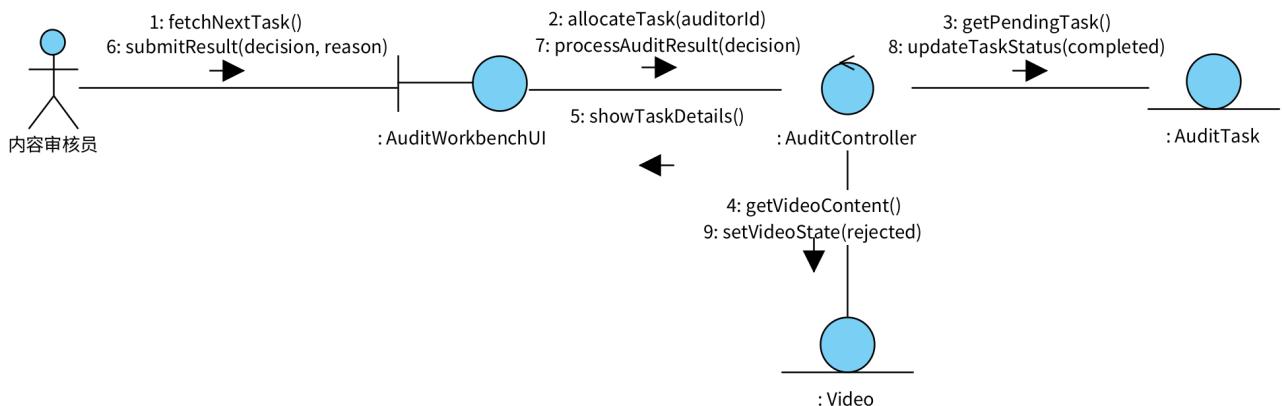


图 4-2 “乐拍视界” 中的审核视频通讯图

### 1. 获取任务：

- 1: `fetchNextTask()`: 内容审核员在 `:AuditWorkbenchUI` 点击“获取任务”。
- 2: `allocateTask(auditorId)`: 界面向 `:AuditController` 请求分配任务。
- 3: `getPendingTask()`: `:AuditController` 从 `:AuditTask` 队列中获取一个待处理的任务。
- 4: `getVideoContent()`: `:AuditController` 根据任务关联，从 `:Video` 获取视频播放链接和描述文本。
- 5: `showTaskDetails()`: `:AuditController` 将内容返回给界面进行展示。

### 2. 提交审核结果：

- 6: `submitResult(decision, reason)`: 审核员观看后，在 `:AuditWorkbenchUI` 提

交决策（如“驳回”及原因）。

7: processAuditResult(decision): 界面将决策数据发送给 :AuditController。

### 3. 更新状态:

8: updateTaskStatus(completed): :AuditController 更新 :AuditTask 的状态为“已结束”。

9: setVideoState(rejected): :AuditController 向 :Video 发送消息，将视频状态更新为“公开”或“违规下架”等。

### 3. 创建发布活动

#### 3.1. 参与对象（协作角色）

1. 运营人员 (Actor): 活动发起者。

2. :OperationsDashboardUI (Boundary): 运营管理后台界面。

3. :ActivityController (Control): 负责活动的创建逻辑和验证。

4. :OperationActivity (Entity): 存储活动规则、时间等信息。

#### 3.2. 通讯图

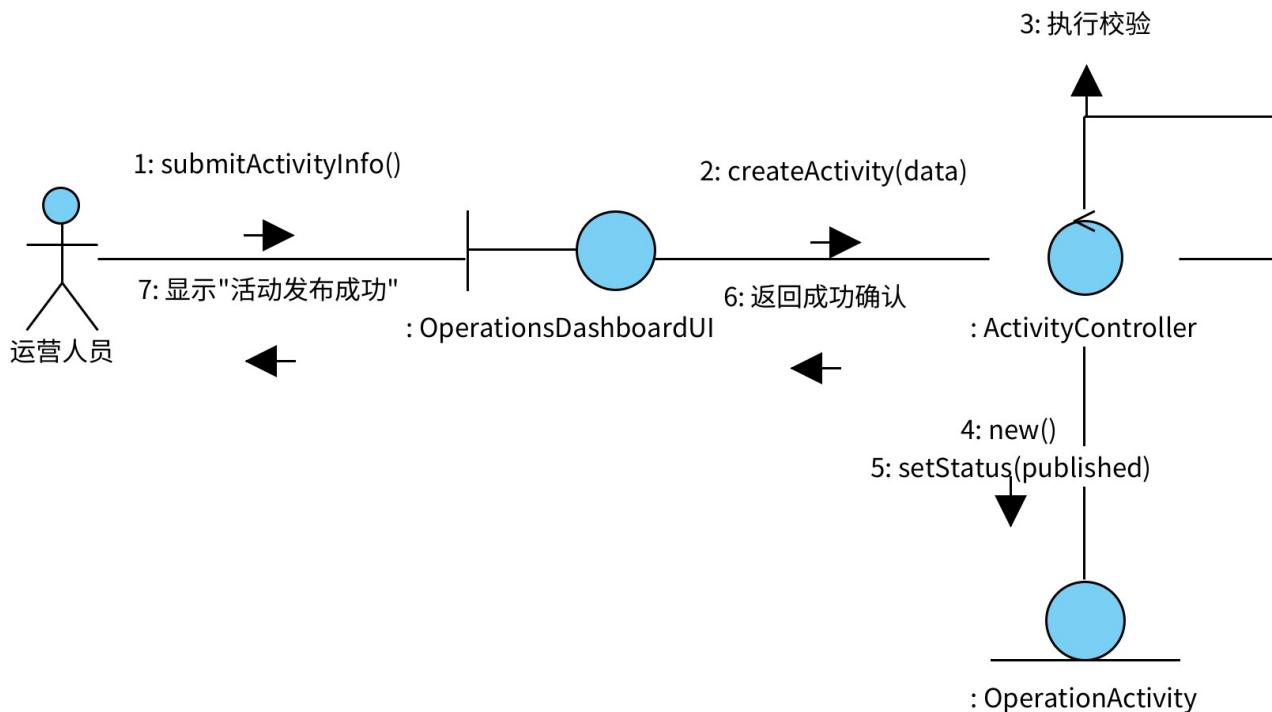


图 4-3 “乐拍视界” 中的创建发布活动通讯图

### 1. 提交活动信息:

运营人员在 :OperationsDashboardUI 输入活动详情（名称、规则、时间），发送 submitActivityInfo() 消息。

## 2. 验证与创建：

:OperationsDashboardUI 将数据传递给 :ActivityController，发送 createActivity(data) 消息。

:ActivityController 执行内部校验逻辑（如检查时间冲突）。

校验通过后，:ActivityController 向 :OperationActivity 发送 new() 创建消息，生成一个新的活动实例。

## 3. 发布与反馈：

:ActivityController 向 :OperationActivity 发送 setStatus(published) 消息，将活动设为生效。

:ActivityController 向 :OperationsDashboardUI 返回成功确认。

界面向运营人员显示“活动发布成功”。

## 4.4. 职责分配与 CRC 卡

为了验证分析类的完整性并合理分配系统职责，我们为核心用况中的关键类创建了 CRC 卡。以下卡片描述了类的高层职责以及为了完成这些职责所需的协作者。

### 1. 浏览个性化视频相关类

#### 1.1. Class Name: FeedController (控制类)

| Responsibilities (职责) | Collaborations (协作) |
|-----------------------|---------------------|
| 协调个性化推荐列表的生成与刷新       | RecommendationModel |
| 获取视频的详细元数据与播放地址       | Video               |
| 获取视频创作者的个人信息（头像、昵称）   | User                |
| 将组装好的视频流数据传递给界面展示     | MainFeedUI          |

#### 1.2. Class Name: RecommendationModel (实体类)

| Responsibilities (职责)   | Collaborations (协作) |
|-------------------------|---------------------|
| 维护推荐算法策略与规则             |                     |
| 根据用户 ID 计算并返回推荐视频 ID 列表 | User                |

| <b>Responsibilities (职责)</b> | <b>Collaborations (协作)</b> |
|------------------------------|----------------------------|
| 接收并处理用户行为反馈以更新兴趣模型           |                            |

### 1.3. Class Name: Video (实体类)

| <b>Responsibilities (职责)</b> | <b>Collaborations (协作)</b> |
|------------------------------|----------------------------|
| 维护视频核心元数据 (URL、标题、描述、标签)     |                            |
| 维护视频的发布状态 (审核中、已发布、已下架)      |                            |
| 提供关联的背景音乐信息                  | Music                      |
| 记录并提供统计数据 (点赞数、评论数、转发数)      |                            |

### 1.4. Class Name: User (实体类)

| <b>Responsibilities (职责)</b> | <b>Collaborations (协作)</b> |
|------------------------------|----------------------------|
| 维护用户基本信息 (ID、昵称、头像、认证状态)     |                            |
| 维护用户的社交关系 (关注列表、粉丝列表)        |                            |
| 维护用户的兴趣画像数据                  |                            |

## 2. 审核用户发布内容相关类

### 2.1. Class Name: AuditController (控制类)

| <b>Responsibilities (职责)</b> | <b>Collaborations (协作)</b> |
|------------------------------|----------------------------|
| 获取并分配待处理的审核任务                | AuditTask                  |
| 获取待审核视频的完整内容供预览              | Video                      |
| 处理审核员提交的决策结果 (通过/驳回)         | AuditWorkbenchUI           |
| 根据审核结果更新视频的可见性状态             | Video                      |
| 记录审核操作日志                     | AuditTask                  |

### 2.2. Class Name: AuditTask (实体类)

| Responsibilities (职责)    | Collaborations (协作) |
|--------------------------|---------------------|
| 维护审核任务队列与状态（待审核、处理中、已结束） |                     |
| 关联具体的待审核视频对象             | Video               |
| 记录负责该任务的审核员信息            | User (Auditor)      |
| 记录审核结束时间与处理意见            |                     |

### 3. 策划并执行运营活动相关类

#### 3.1. Class Name: ActivityController (控制类)

| Responsibilities (职责) | Collaborations (协作)           |
|-----------------------|-------------------------------|
| 验证活动规则与时间的冲突性         |                               |
| 创建并初始化新的运营活动实例        | OperationActivity             |
| 发布活动并将其推送到前台          | OperationActivity, MainFeedUI |
| 收集活动参与数据并生成报告         |                               |

#### 3.2. Class Name: OperationActivity (实体类)

| Responsibilities (职责) | Collaborations (协作) |
|-----------------------|---------------------|
| 维护活动详情（名称、规则、起止时间）    |                     |
| 维护活动状态（草稿、进行中、已结束）    |                     |
| 聚合参与该活动的视频列表          | Video               |

### 4.5. 整合分析类图

通过对“浏览个性化视频流”、“审核用户发布内容”及“策划并执行运营活动”等核心用况的分析，我们将局部产生的类图进行整合，形成了“乐拍视界”的整体分析类图。该模型展示了系统逻辑架构中的静态结构。

#### 1. 实体类 (Entity Classes)

##### 1.1. Video (短视频)

属性: videoID, title, url, createTime, status (审核中/已发布/已下架), likeCount, commentCount

操作: getDetails(), updateStatus(newStatus), incrementStats(type)



图 4-4 “乐拍视界” 中的 Video 类图

### 1.2. User (用户)

属性: userID, nickname, avatarUrl, role (普通用户/审核员/运营), interestTags

操作: getInfo(), updateProfile(data)



图 4-5 “乐拍视界” 中的 User 类图

### 1.3. Music (音乐)

属性: musicID, title, artist, audioUrl, copyrightStatus

操作: getMusicInfo()



图 4-6 “乐拍视界” 中的 Music 类图

### 1.4. AuditTask (审核任务)

属性: taskID, createTime, auditStatus (待处理/通过/驳回), rejectReason

操作: assignTo(auditorID), updateResult(decision, reason)

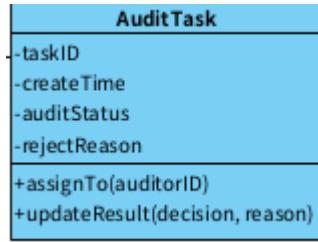


图 4-7 “乐拍视界” 中的 AuditTask 类图

### 1.5. OperationActivity (运营活动)

属性: activityID, name, rules, startTime, endTime, state

操作: publish(), addVideo(videoID)

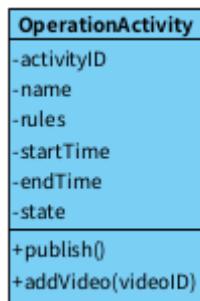


图 4-8 “乐拍视界” 中的 OperationActivity 类图

## 2. 控制类 (Control Classes)

1. FeedController: refreshFeed(userId), loadMore()

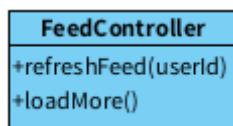


图 4-9 “乐拍视界” 中的 FeedController 类图

2. AuditController: getPendingTask(), submitAuditResult(taskID, decision)

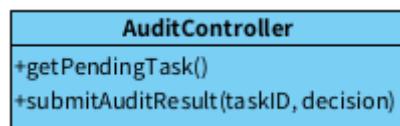


图 4-10 “乐拍视界” 中的 AuditController 类图

3. ActivityController: createActivity(info), publishActivity(activityID)

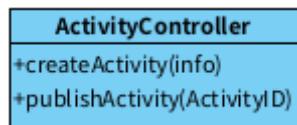


图 4-11 “乐拍视界” 中的 ActivityController 类图

### 3. 边界类 (Boundary Classes)

MainFeedUI: displayVideos(list), showError(msg)



图 4-12 “乐拍视界” 中的 MainFeedUI 类图

AuditWorkbenchUI: showTask(videoData), showSuccess(msg)

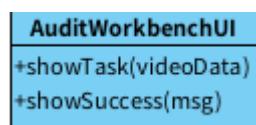


图 4-13 “乐拍视界” 中的 AuditWorkbenchUI 类图

### 4. 类关联关系

#### 4.1. 用户与视频 (发布关系)

关联: User —— Video

多重性: 1 (User) <---> 0..\* (Video)

说明: 一个用户可以发布零个或多个视频; 一个视频必须属于且仅属于一个创作者。

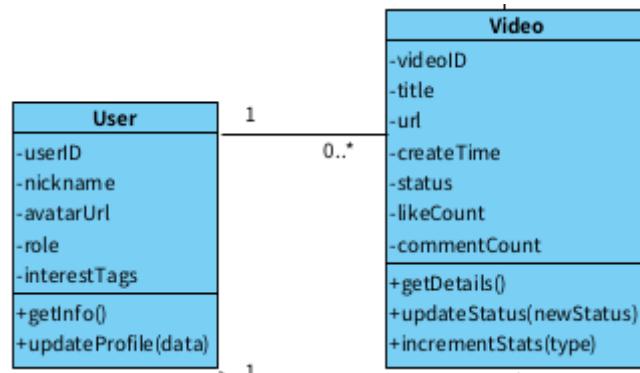


图 4-14 “乐拍视界” 中的用户与视频 (发布关系)图

#### 4.2. 视频与音乐 (使用关系)

关联: Video —— Music

多重性: 0..\* (Video) <---> 0..1 (Music)

说明: 一个视频可以使用零首 (原声) 或一首背景音乐; 一首音乐可以被多个视频使用。

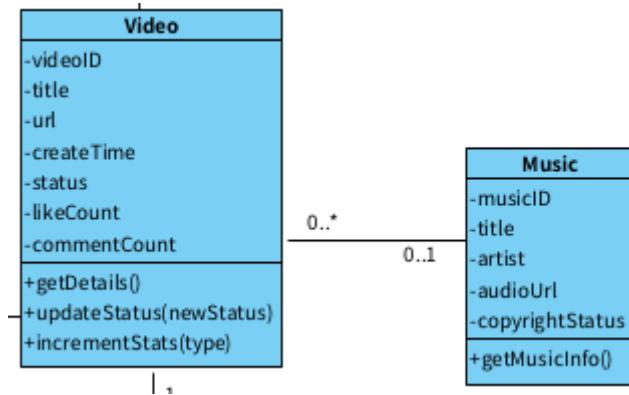


图 4-15 “乐拍视界” 中的视频与音乐 (使用关系) 图

#### 4.3. 审核任务与视频 (审核对象关系)

关联: AuditTask —— Video

多重性: 0..1 (AuditTask) <---> 1 (Video)

说明: 一个审核任务针对且仅针对一个视频; 一个视频在特定时间点关联零个或一个活跃的审核任务。

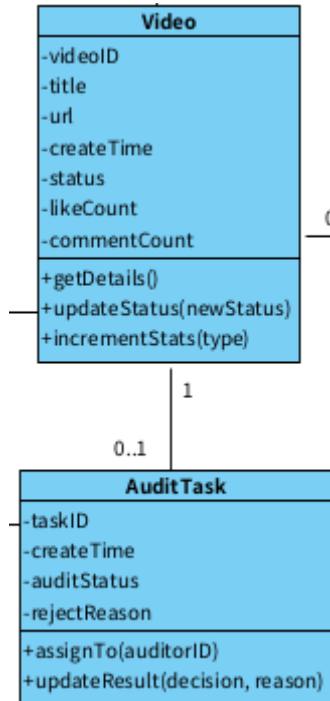


图 4-16 “乐拍视界” 中的审核任务与视频 (审核对象关系) 图

#### 4.4. 审核任务与用户 (执行关系)

关联: User (Auditor) —— AuditTask

多重性: 1 (User) <---> 0..\* (AuditTask)

说明: 一个审核员可以处理多个审核任务; 一个具体的审核任务由一名审核员负责。

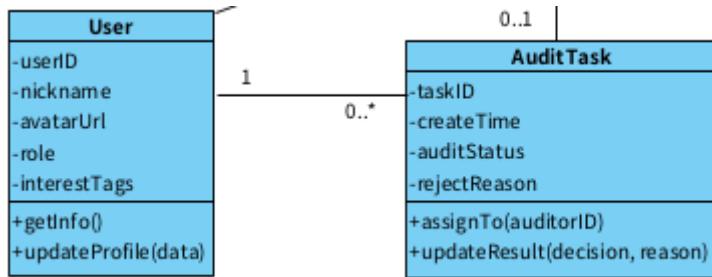


图 4-17 “乐拍视界” 中的审核任务与用户 (执行关系)图

#### 4.5. 运营活动与视频 (聚合关系)

关联: OperationActivity — Video

多重性: 0..1 (OperationActivity) <---> 0..\* (Video)

说明: 一个活动包含多个参与视频; 一个视频可以不参与活动或参与一个活动。

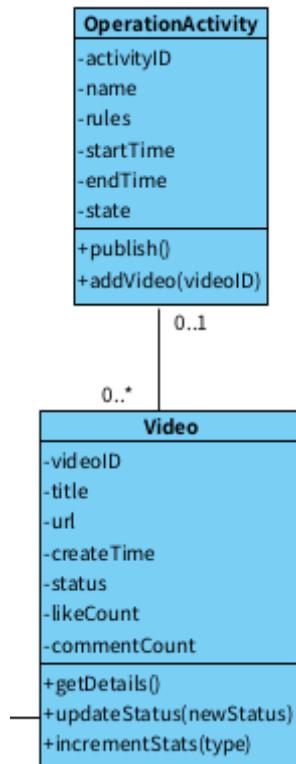


图 4-18 “乐拍视界” 中的运营活动与视频 (聚合关系)图

## 第5章 架构设计

### 5.1. 设定权衡优先级

为了确保系统架构能够最大限度地支持产品的核心价值——即“极致流畅的视听体验”与“活跃的社区互动”，我们基于商业目标和非功能性需求，制定了以下设计目标的优先级指导原则。

#### 1. 核心设计目标的优先级排序

我们确立了“性能与可用性优先，可扩展性次之，适度牺牲数据强一致性与初期成本”的总体架构策略。具体优先级排序如下：

##### 第一优先级（最高权重）：性能与可用性

**定义：**系统必须对用户的操作做出即时响应。视频播放必须达到“秒开”标准，滑动切换无卡顿；视频拍摄与特效渲染必须实时完成，且界面交互必须直观、流畅。

**理由：**“乐拍视界”面向的是耐心极低、追求感官刺激的年轻用户群体。任何显著的延迟或交互上的阻碍都会直接导致用户流失。

##### 第二优先级（高权重）：可扩展性与安全性

**定义：**架构必须支持从初期的种子用户快速平滑扩展至百万级日活用户；同时必须保障用户隐私数据安全及内容合规。

**理由：**短视频产品具有爆发性增长的特征，系统必须能够通过增加硬件资源来线性提升处理能力（水平扩展），以应对突发流量。同时，内容合规是平台长期运营的红线。

##### 第三优先级（中权重）：可维护性与上市时间

**定义：**代码结构应清晰，便于后续迭代；系统需在6-12个月内完成核心功能上线。

**理由：**虽然长期维护很重要，但在激烈的市场竞争初期，抢占市场窗口期更为关键。我们可以接受在初期引入少量的“技术债务”，以换取核心功能的快速交付。

##### 第四优先级（低权重）：数据强一致性与运营成本

**定义：**各个节点的数据必须在同一时刻保持绝对一致；硬件和带宽资源的节省。

**理由：**在社交场景下，用户对点赞数、浏览量的实时精确性不敏感，因此可以采用“最终一致性”模型。此外，为了保障第一优先级的“性能”，我们愿意在初期投入较高的带宽和CDN成本。

## 2. 具体的架构权衡决策

基于上述优先级排序，我们在系统设计的关键领域做出了以下具体的权衡决策：

### 2.1. 空间换时间

决策描述：为了满足高性能的视频播放需求，我们选择消耗更多的存储空间和内存资源。

具体措施：

多版本转码存储：服务器端不仅存储视频原片，还预先转码生成多种分辨率、多种码率的视频副本。

激进的缓存策略：在移动端本地和 CDN 节点大量使用缓存技术。

### 2.2. 可用性优于一致性

决策描述：根据 CAP 理论，在分布式系统中，面对网络分区时，我们优先保障可用性。

具体措施：

最终一致性设计：对于点赞、评论计数、粉丝数等高频并发数据，系统不保证所有用户在同一毫秒看到的数据是完全一致的。我们允许数据在短时间内存在差异，通过异步消息队列进行后端处理，确保数据在最终状态下是一致的。

降级策略：当推荐算法服务不可用或响应超时时，系统将自动降级为加载本地缓存或通用的热门列表。

### 2.3. 功能分级与快速迭代

决策描述：为了满足上市时间的要求，我们对功能进行严格分级，牺牲部分非核心功能的上线时间。

具体措施：

MVP（最小可行性产品）策略：首个版本集中资源打磨“拍摄-剪辑-发布-浏览”的核心闭环。对于复杂的“即时通讯（IM）高级功能”、“多级分销系统”等，推迟到后续版本迭代。

使用现成而非自研：对于非核心竞争力的功能模块（如美颜算法库、即时通讯底层通道），优先使用成熟的第三方服务，而不是投入团队自研，以缩短开发周期。

### 2.4. 动态性优于静态效率

决策描述：为了支持推荐算法的实时反馈，我们选择动态内容分发机制。

具体措施：

**实时流式计算：**架构将支持基于用户实时行为（滑过、完播、点赞）的实时流式计算推荐。

### 3. 架构决策的验证与调整

系统架构师将负责监控上述权衡决策在实际开发和运行中的效果。如果发现某项低优先级的指标增长失控并威胁到项目生存，或者市场环境发生变化，架构师将及时发起架构评审，动态调整优先级策略。当前的优先级设定旨在确保“乐拍视界”能够作为一个高性能、高粘性的社交平台顺利推向市场。

## 5.2. 估算系统性能

在系统架构设计的早期阶段，我们基于“乐拍视界”的中期业务目标——即日活跃用户(DAU)达到100万的规模，对系统的核心性能指标进行了粗略估算。考虑到互联网流量的突发性，我们在计算结果的基础上增加了2至3倍的冗余量，以确保架构足以应对高峰负载和营销活动带来的流量冲击。

### 1. 基础假设与模型参数

为了进行有效的估算，我们设定了以下基础业务模型参数：

日活跃用户(DAU)：1,000,000人。

用户平均在线时长：30分钟。

平均观看视频数：假设每位用户每天观看30个短视频（含完整观看与快速划过）。

内容生产比例：假设1%的用户每天上传1个视频（即每天新增10,000个视频）。

互动率：假设用户对观看的视频中有5%进行点赞操作。

峰值时段：假设全天80%的流量集中在4小时的晚间高峰期(20:00 - 24:00)。

### 2. 计算吞吐量

#### 2.1. 视频浏览

日总请求量： $1,000,000 \text{ 用户} \times 30 \text{ 视频} = 30,000,000 \text{ 次请求/天}$ 。

平均QPS： $30,000,000 / (24 \times 3600) \approx 347 \text{ QPS}$ 。

峰值QPS估算：

根据二八原则与峰值时段假设： $(30,000,000 \times 0.8) / (4 \times 3600) \approx 1,666 \text{ QPS}$ 。

架构设计目标QPS：考虑到突发热点，乘以3倍安全系数，系统需支持5,000 QPS的视频流获取请求。

结论：单台应用服务器难以承担，必须采用负载均衡集群，且核心推荐接口需具备毫秒级响应能力。

## 2.2. 视频上传

日总上传量：10,000 个视频/天。

峰值上传 TPS：假设上传也集中在高峰期， $(10,000 \times 0.8) / (4 \times 3600) \approx 0.6$  TPS。

架构设计目标 TPS：即使考虑到活动期间上传量翻倍，设计目标定为 5-10 TPS。

结论：虽然 TPS 数值不高，但单次上传耗时长、占用带宽大且触发后续转码流程。架构上必须采用异步处理，即上传完成后立即响应用户，转码在后台排队进行。

## 2.3. 点赞视频

日总互动量：30,000,000 浏览  $\times 5\% = 1,500,000$  次互动/天。

峰值 TPS： $(1,500,000 \times 0.8) / 14,400 \approx 83$  TPS。

架构设计目标 TPS：乘以 3 倍冗余，设计目标约为 250 - 300 TPS。

结论：写入压力尚可，但考虑到点赞需要实时反馈给前端，使用 Redis 等高性能缓存数据库来处理计数，随后异步持久化到关系型数据库。

## 3. 估算存储需求

单个视频大小：假设原视频平均 20MB，经过系统转码生成不同清晰度版本后，总存储占用约为 30MB。

日新增存储量：10,000 视频/天  $\times 30\text{MB} = 300,000\text{ MB} \approx 300\text{ GB/天}$ 。

年存储需求： $300\text{ GB} \times 365 \approx 109.5\text{ TB/年}$ 。

结论：

存储增长迅速，无法使用单机硬盘存储。

架构必须采用分布式对象存储服务，支持无限水平扩展。

需要设计冷热数据分离策略，1 年以上的旧视频可归档至低成本存储。

## 4. 估算网络带宽

平均视频码率：假设流畅播放需要的平均下行流量为 5MB/视频。

峰值流量消耗：

峰值时刻每秒有 1,666 个视频被请求播放。

每秒所需带宽流量 =  $1,666 \times 5\text{MB} \approx 8,330\text{ MB/s}$ 。

换算为带宽速率 =  $8,330\text{ MB/s} \times 8\text{ bits} \approx 66\text{ Gbps}$ 。

结论：

66 Gbps 的瞬时流量对于自建机房是巨大的压力且成本极高。

架构决策：引入内容分发网络（CDN）。核心服务器仅负责调度和业务逻辑，95% 以上的视频流流量应由 CDN 边缘节点承担。

## 5. 性能估算总结与架构调整

基于上述粗略计算，我们对系统架构做出以下针对性确认：

硬件资源配置：数据库服务器不需要极其昂贵的顶配硬件，因为读写 QPS 在 RDBMS 的处理范围内，但推荐系统计算节点需要高性能 CPU/GPU 支持。

瓶颈预判：系统的主要瓶颈不在于应用服务器的并发数，而在于网络带宽和存储 I/O。

架构修正：

读写分离：鉴于读（浏览）远大于写（上传），数据库应设计为主从复制架构，多台从库负责读取。

静态资源剥离：所有视频、图片等静态资源强制走 CDN，严禁直接从应用服务器读取。

缓存策略：由于 80% 的播放集中在 20% 的热门视频上，必须在各级（客户端、CDN、服务端）建立激进的缓存策略以降低回源流量。

### 5.3. 选择架构风格

为了满足“乐拍视界”在高性能互动、海量内容处理及多端适配方面的需求，我们采用混合架构风格。

#### 1. 整体系统架构：分布式客户/服务器风格

在宏观层面，系统采用客户/服务器架构，明确划分前端（移动设备）与后端（云端集群）的职责边界。

客户端（胖客户端策略）：我们采用“胖客户端”模式。移动端 App 不仅仅是信息的展示层，还承担了大量的计算任务（如美颜滤镜渲染、视频压缩），以利用用户设备的计算能力，减少服务器压力并降低网络延迟带来的体验损耗。

服务器端：后端作为一个逻辑上的整体服务器，负责数据持久化、复杂业务逻辑处理

(如推荐算法计算)、多用户间的社交关系维护以及全局资源的协调。

通信机制：客户端与服务器之间采用 RESTful API 进行通信。

## 2. 客户端架构：MVC 风格

为了应对移动端高度复杂的交互逻辑（如拍摄界面与编辑界面的切换、信息流的滑动与点赞），客户端内部采用了 MVC (Model-View-Controller) 架构风格，以实现用户界面与业务逻辑的分离。

模型 (Model)：封装核心数据与状态。模型负责在数据发生变化时（如收到新的点赞通知）通过观察者机制通知视图。

视图 (View)：负责数据的可视化呈现。视图不包含业务逻辑，仅根据模型的数据进行渲染。

控制器 (Controller)：处理用户的输入事件。

优势：这种分离使得我们可以在不修改核心逻辑的情况下，轻松替换 UI 主题，或针对不同尺寸的屏幕开发不同的视图。

## 3. 服务器端架构：分层与分区结合

服务器端采用封闭的分层架构 (Closed Layered Architecture)，结合 垂直分区 (Partitioning) 策略。

系统自上而下划分为四个标准层次，上层仅依赖于其直接下层：

表示层：

职责：处理来自客户端的 HTTP/HTTPS 请求，进行参数校验、身份认证 (Token 验证)，并将请求分发给应用逻辑层。

组件：API 网关、请求控制器。

应用逻辑层：

职责：编排业务流程，协调领域对象完成具体任务。

组件：VideoService、UserService、NotificationService。

领域层：

职责：包含核心业务规则和状态，与具体应用场景无关。例如，判断“视频时长是否超限”。

组件：实体类、值对象、领域服务。

数据持久层 / 基础设施层：

职责：负责与数据库、缓存、文件存储系统进行交互，隐藏具体的技术实现细节（如 SQL 语句）。

组件：DAO（数据访问对象）、ORM 映射器。

为了支持并行开发和未来的微服务化拆分，我们在每一层内部进行了垂直分区，将系统划分为若干个弱耦合的子系统：

用户子系统：处理注册、登录、个人资料管理。

内容子系统：处理视频的上传、存储、元数据管理。

社交子系统：处理关注、粉丝、评论、私信。

推荐子系统：处理用户画像分析、内容分发。

#### 4. 视频处理子系统：管道-过滤器风格

针对视频上传后的后端处理流程，我们选用了管道-过滤器 (Pipe-and-Filter) 架构风格。

输入：原始视频文件流。

过滤器 (Filters)：

格式校验过滤器：检查文件头，确保是合法的视频格式。

转码过滤器：将视频转换为不同分辨率和编码格式，以适配不同网络环境。

抽帧过滤器：截取视频关键帧作为封面图。

水印过滤器：添加“乐拍视界”品牌水印。

管道 (Pipes)：数据流在过滤器之间传递的通道。

优势：支持增量开发和并行处理。

#### 5. 数据存储架构：主从复制与读写分离

考虑到系统“读多写少”（浏览量远大于上传量）的特性，数据存储架构采用了主从复制 (Master-Slave Replication) 风格。

主库：负责处理所有的写入、更新和删除操作。

从库：多个从库负责处理读取操作。

机制：主库的数据变更会近乎实时地同步到从库，从而实现负载均衡，提高系统的并发读取能力。

## 5.4. 将系统划分成子系统

根据物理部署边界将系统划分为两大顶级子系统：移动客户端子系统与云端服务子系统。并在各自的内部，依据功能职责（分区）和抽象级别（分层）进一步细化为具体的逻辑子系统。

### 1. 移动客户端子系统的划分

移动客户端承担着内容生产与消费的双重职能，其内部架构主要依据功能分区策略进行划分，以支持富媒体处理与复杂的交互逻辑。

#### 1.1. 媒体采集与处理子系统

职责：负责底层的音视频数据流处理。

核心功能：

拍摄控制：调用摄像头硬件，管理对焦、曝光、分辨率。

实时渲染：对视频流进行逐帧处理，应用美颜、瘦脸、滤镜算法（基于OpenGL/Metal）。

非线性编辑：实现多段视频剪辑、转场拼接、时间特效（慢动作/倒放）。

音频合成：将背景音乐（BGM）与原声进行混音，并实现“智能节拍对齐”。

接口：提供 RecorderInterface 和 EditorInterface 供 UI 层调用。

#### 1.2. 播放与互动子系统

职责：负责沉浸式信息流的展示与用户交互。

核心功能：

流媒体播放：视频解码、缓冲管理、自适应码率切换（HLS/DASH）。

手势交互：捕获双击点赞、滑动手势，并将其转换为业务指令。

#### 1.3. 本地数据管理子系统

职责：管理客户端的持久化数据，确保在弱网或离线环境下的可用性。

核心功能：

草稿箱管理：存储未发布的视频工程文件（断点保护）。

缓存管理：对已加载的视频流、图片进行 LRU 缓存策略管理，节省流量。

用户配置：存储登录 Token、用户偏好设置。

### 2. 云端服务子系统的划分

云端服务采用分层架构作为基础骨架，在应用逻辑层和领域层采用垂直分区策略，拆分为以下独立的业务子系统。

## 2.1. 用户中心子系统

职责：维护全平台的用户身份与画像。

服务内容：注册/登录（支持手机号、第三方授权）、实名认证、个人档案管理。

对外接口：UserService（提供用户鉴权、基本信息查询）。

## 2.2. 视频内容管理子系统

职责：处理视频全生命周期的管理。

服务内容：

上传调度：接收客户端上传的视频文件，分配对象存储地址。

转码服务：将原视频异步转码为720P/1080P等多版本，生成封面图。

元数据管理：存储视频标题、描述、音乐引用信息。

对外接口：VideoService（提供视频发布、详情查询、删除接口）。

## 2.3. 社交互动子系统

职责：维护人与人、人与内容之间的关系链。

服务内容：

关系链管理：关注、粉丝列表维护。

互动行为：点赞计数与状态记录。

对外接口：SocialService（提供关注操作、获取互动状态接口）。

## 2.4. 智能推荐子系统

职责：系统的“大脑”，决定用户看到什么内容。

服务内容：

离线计算：基于用户历史行为建立兴趣模型。

在线排序：实时从候选池中召回视频，并根据预测点击率（CTR）进行精细化排序。

冷启动处理：为新用户和新视频提供试探性流量策略。

对外接口：FeedService（输入用户ID，输出推荐视频列表）。

## 2.5. 内容安全与审核子系统

职责：系统的“守门人”，确保内容合规。

服务内容：

人工复审工作台：为审核人员提供可疑视频的分发与处理界面。

通信方式：这是一个典型的事件驱动子系统，通过消息队列监听“视频上传成功”事件。

## 3. 子系统间的依赖与通信关系

通信风格：

客户端与服务端：采用标准的 HTTP/RESTful API 进行同步通信。

服务端子系统间：

强依赖场景采用 RPC（远程过程调用），以保证低延迟。

解耦场景采用消息队列进行异步通信，实现削峰填谷。

依赖原则：

上层子系统可以依赖下层基础子系统。

同层子系统之间尽量避免双向循环依赖，如需交互，通过定义独立的接口层或事件总线进行解耦。

## 4. 物理部署映射

客户端子系统部署于用户的 Android 智能手机终端。

云端业务子系统（用户、内容、社交）部署于应用服务器集群（Kubernetes Pods）。

推荐子系统部署于配备 GPU 的高性能计算集群。

数据存储依赖于独立的数据库集群（PostgreSQL）、缓存集群（Redis）和对象存储服务（OSS）。

## 5.5. 确定问题内部的并发性

为了确保“乐拍视界”在百万级日活下的高性能表现以及移动端的极致流畅体验，我们通过分析系统用况和对象生命周期，识别出以下关键的并发活动区域，并定义了相应的控制线程管理策略。

### 1. 移动客户端的并发设计

移动端的并发设计核心目标是“UI 主线程永不阻塞”，确保用户在进行重资源操作（如

视频导出、上传) 时，界面依然流畅响应。我们识别出以下必须分离的独立控制线程：

UI 渲染线程 (主线程)：

职责：仅负责界面绘制、响应用户的点击与滑动手势、更新进度条显示。

约束：该线程禁止执行任何超过 16ms 的耗时操作 (如网络请求、数据库读写、图像处理)，以保证 60fps 的帧率。

媒体采集与处理线程 (工作线程组)：

视频流处理线程：负责从摄像头硬件获取原始 YUV 数据流，并调用 GPU 进行实时的滤镜渲染和美颜处理。

音频编码线程：独立负责麦克风数据的采集与 AAC 编码，必须与视频线程保持高精度的时间戳同步。

合成写入线程：当用户点击“录制”时，该线程将处理后的音视频流复用 (Mux) 并写入本地文件系统。

网络通信线程 (异步 I/O)：

场景描述：当用户处于弱网环境浏览视频流时，网络请求可能耗时较长。

并发策略：所有的 HTTP API 请求 (如获取推荐列表、点赞) 和 CDN 资源加载均在独立的网络线程池中执行。回调函数在数据获取成功后，通过消息机制通知 UI 线程刷新界面。

后台上传任务 (守护线程)：

并发对象：UploadManager。

生命周期：当用户点击“发布”后，该对象在后台服务中独立运行。即使用户切换到“浏览”页面或退出 App 到后台，视频的分片上传、断点续传逻辑仍需持续执行，直至任务完成或失败。

## 2. 服务端的并发设计

服务端面临高并发请求压力，无法采用简单的单线程模型。我们将并发处理分为“同步请求处理”和“异步后台处理”两个维度。

请求处理并发 (Thread-per-Request 模型)：

Web 容器线程池：应用服务器 (如基于 Tomcat 或 Netty) 维护一个受控的线程池。对于每一个来自客户端的 HTTP 请求 (如“刷新首页”)，系统分配一个独立的线程进行

处理。

**数据库连接池：**为了支持多线程并发访问数据库，系统维护数据库连接池。多个业务线程可以同时借用连接进行查询或写入，互不干扰，从而实现数据访问层面的并发。

**视频处理流水线（Pipeline Concurrency）：**

**问题识别：**视频转码、封面截取均属于 CPU/GPU 密集型且耗时极长的操作。如果在请求线程中同步执行，将导致服务器线程耗尽。

**并发策略：**采用事件驱动的并发模型。

**生产者：**上传服务接收视频后，仅将“视频已上传”的消息写入消息队列（如 Kafka），随即释放请求线程。

**消费者：**后端维护多个独立的消费者服务集群（转码服务）。这些服务并行工作，互不阻塞。

### 3. 对象状态与互斥管理

在并发环境中，必须管理共享资源的访问冲突，以保证数据一致性。

**全局计数器的并发控制：**

**场景：**热门视频可能在同一毫秒内收到数千个“点赞”请求。

**策略：**避免直接并发更新数据库行。采用 Redis 的原子递增操作（Atomic Increment）在内存中处理并发计数，随后通过异步任务批量持久化到数据库。

**用户会话的互斥：**

**场景：**同一用户账号可能在多台设备同时登录并操作。

**策略：**采用分布式锁或 Token 版本控制机制。当检测到关键状态变更（如修改密码）时，强制使其他并发的旧 Token 失效。

### 4. 边界条件的并发处理

**资源竞争处理：**当多个上传任务同时进行时，UploadManager 将根据网络带宽情况，限制最大并发上传数为 1-2 个，其余任务进入等待队列。

**死锁预防：**在涉及跨子系统调用的复杂事务中（如同时更新社交关系和消息通知），严格规定资源锁定的顺序，并设置超时释放机制。

### 5.6. 配置子系统的硬件

根据系统各组件对计算资源（CPU/GPU）、内存、存储及网络带宽的不同需求，我们

将“乐拍视界”的物理架构划分为五类核心硬件节点，并定义了相应的网络连接拓扑。

## 1. 物理节点分类与配置

我们采用云服务器集群配置。

### 1.1. 移动终端节点

部署子系统：媒体采集与处理子系统、播放与互动子系统、本地数据管理子系统。

硬件特征：

设备类型：用户持有的 Android 智能手机。

关键资源：高清摄像头与麦克风阵列（用于采集）、GPU/NPU（用于本地实时的美颜渲染与特效处理）、本地闪存（用于草稿箱存储）。

配置要求：系统向下兼容至 Android 7.0 及主流中低端机型，但在高端机型上开启高帧率（60fps）录制与高清播放模式。

### 1.2. 应用服务计算节点

部署子系统：用户中心子系统、社交互动子系统、以及视频内容管理子系统的业务逻辑部分。

硬件特征：

实例类型：通用计算型云服务器。

配置规格：配置 4 vCPU / 8GB RAM。

扩展策略：部署于负载均衡器之后，配置自动伸缩组。根据 CPU 利用率（阈值 70%），集群规模可从初始的 1 台动态扩展至 50+ 台，以应对流量潮汐。

### 1.3. 媒体处理与智能计算节点

部署子系统：智能推荐子系统、内容安全与审核子系统、视频转码服务。

硬件特征：

实例类型：异构计算加速型服务器（GPU Instances）。

关键资源：配备 NVIDIA T4 或 A10 Tensor Core GPU。

配置理由：推荐算法的深度学习推理（Inference）以及视频的高速转码、抽帧、图像识别属于计算密集型任务，单靠 CPU 处理效率过低，必须利用 GPU 加速以保证毫秒级的处理时延。

#### 1.4. 数据存储节点

部署子系统：数据库集群（PostgreSQL）、缓存集群（Redis）。

硬件特征：

实例类型：内存优化型与高 I/O 型服务器。

配置规格：

缓存节点：16GB - 32GB RAM，用于存储热点视频列表与 Session 数据，确保极高的读写吞吐量。

数据库节点：配备 NVMe SSD 高速固态硬盘，保障高并发下的事务写入性能。

#### 1.5. 边缘分发节点

部署子系统：静态资源缓存（视频切片文件、封面图、头像）。

硬件特征：

分布位置：遍布全国各省市及主要运营商的边缘机房。

作用：承担系统 95% 以上的出网带宽流量，使用户能就近获取视频流，最大限度降低播放卡顿率。

### 2. 网络连接与拓扑结构

为了保障数据传输的效率与安全性，硬件节点之间的连接遵循以下拓扑规则：

外部接入网络：

移动终端节点通过 4G/5G/Wi-Fi 网络连接互联网。

入口流量经过 DNS 解析与 CDN 调度，静态请求直达边缘节点，动态 API 请求通过 HTTPS (TLS 1.3) 加密通道路由至云端数据中心的负载均衡器。

内部私有网络 (VPC)：

所有云端服务器节点（应用、计算、存储）均部署在隔离的虚拟私有云 (VPC) 内部。

节点间通过 10Gbps - 25Gbps 的高速内网互联，确保微服务调用（RPC）和数据库访问的低延迟。

安全组策略：数据存储节点不分配公网 IP，仅允许应用服务节点通过特定端口访问，物理隔离外部攻击。

### 3. 硬件冗余与故障转移

为了满足高可用性需求，硬件配置实施了冗余策略：

多可用区部署 (Multi-AZ)：所有核心服务节点和数据节点必须跨越至少两个物理隔离的可用区进行部署。当一个机房发生电力或网络故障时，流量自动切换至另一可用区，确保服务不中断。

主从热备：数据库节点采用“一主多从”配置，主节点故障时，监控系统自动将从节点提升为主节点。

## 5.7. 管理数据存储

“乐拍视界”涉及海量非结构化媒体数据与高并发结构化业务数据，我们构建多级数据存储架构，对不同类型的数据实施分类管理。

### 1. 非结构化数据存储：分布式对象文件系统

存储选型：采用分布式对象存储服务

管理策略：

存储结构：采用扁平化的桶 (Bucket) 与键 (Key) 管理。视频文件的 Key 生成策略采用 Hash(VideoID) + Timestamp。

元数据分离：文件系统中仅存储实际的二进制媒体文件 (Blob)。文件的元属性 (如文件名、大小、上传者 ID、CDN 访问 URL) 存储于关系型数据库中，通过 URL 链接实现逻辑关联。

生命周期管理：实施冷热数据分层。发布超过 12 个月且访问量极低的视频，自动迁移至低成本的归档存储区 (Archive Storage)。

### 2. 结构化核心数据存储：关系型数据库

对于用户账户、视频元信息、社交关系等需要严格数据一致性 (ACID) 和复杂关联查询的业务数据，采用关系型数据库进行管理。

存储选型：PostgreSQL 集群（使用 InnoDB 存储引擎）。

数据模型设计：

用户域：User 表存储用户基本信息，Auth 表存储密码哈希与授权令牌。

内容域：Video 表存储视频的描述、状态、时长及指向 OSS 的文件链接；Music 表存储音乐版权信息。

社交域：Follow 表记录关注关系；Comment 表存储评论内容。

**性能优化策略：**

**读写分离：**部署主从复制架构。主库负责处理“发布视频”等写入事务；多个从库通过负载均衡处理“拉取视频列表”等高频读取请求。

**分库分表（Sharding）：**针对 Video 表和 Comment 表，预计数据量将迅速突破单表亿级。设计基于 UserID 或 VideoID 的哈希分片策略，将数据水平拆分至多个物理数据库实例中，以突破单机 I/O 瓶颈。

### 3. 高频互动数据存储：内存数据库

点赞数、播放量等数据具有极高的并发写入特性，且对数据的实时性要求高于持久性。

**存储选型：**Redis（Key-Value 存储）。

**管理策略：**

**计数器管理：**使用 Redis 的原子递增操作（INCR）处理视频的点赞数和浏览量。系统每隔 N 秒（如 10 秒）将内存中的计数器状态异步批量持久化（Write-Back）到 PostgreSQL 数据库，以减少对磁盘数据库的写入压力。

**会话存储：**用户的 Session Token 和在线状态存储于 Redis 集群，支持分布式环境下的快速鉴权。

### 4. 大数据日志存储：列式数据库

为了支持推荐算法的精准计算，系统需要记录海量的用户行为日志（如“用户 A 在第 5 秒划走了视频 B”）。此类数据写入量巨大且只需追加。

**存储选型：**HBase 或 Cassandra。

**管理策略：**采集端将行为日志写入消息队列，经清洗后存入列式数据库。推荐引擎定期从此处批量读取数据进行离线模型训练。

### 5. 数据访问层的抽象设计

为了降低业务逻辑层与具体存储技术之间的耦合度，在架构中引入了数据访问对象（DAO）模式。

**封装实现：**业务逻辑层（如 VideoService）不直接编写 SQL 语句或操作文件流，而是通过 VideoDAO 接口进行调用。

**灵活性：**确保了当底层存储介质发生变更时，只需修改 DAO 层的实现代码，而无需变动上层业务逻辑。

## 5.8. 处理全局资源

在“乐拍视界”的分布式架构中，多个子系统和并发进程需要共享有限的系统资源。为了保证系统的稳定性和数据的一致性，必须掌控全局资源，并建立严格的访问协议。

### 1. 全局唯一标识符（UUID）生成策略

系统采用去中心化的 ID 生成策略，以确保逻辑实体的全局唯一性和有序性。

生成算法：采用雪花算法（Snowflake Algorithm）的变种。

结构定义：生成 64 位长整型 ID。包含：1 位符号位 + 41 位毫秒级时间戳 + 10 位机器 ID（区分不同机房和服务器）+ 12 位序列号（支持同一毫秒内生成 4096 个 ID）。

应用范围：所有的核心业务对象，包括 UserID、VideoID、CommentID 和 MessageID。

优势：ID 随时间单调递增，有利于数据库索引性能；生成过程在本地内存完成，无网络 I/O 开销，满足高性能需求。

物理文件命名空间：

对于存储在对象存储（OSS）中的视频和图片文件，不直接使用文件名作为标识。

命名规范：采用哈希打散的路径结构。

### 2. 访问控制与身份认证资源

系统通过统一的认证授权中心来管理对所有受保护资源的访问。

身份凭证管理：

采用 JWT (JSON Web Token) 作为全局通用的身份令牌。用户登录后，服务器签发包含 UserID、Role（角色）及过期时间的加密 Token。

无状态验证：客户端在后续的所有 HTTP 请求头中携带该 Token。API 网关层负责解析和验证 Token 的合法性，后端服务不再存储 Session 状态。

逻辑分区与锁定：

资源锁：针对特定的高并发写操作，引入分布式锁（基于 Redis 实现）。

机制：将具体的逻辑资源映射为内存中的一个 Key。线程必须先通过 SETNX 命令争抢该 Key 的持有权，才能操作对应的数据库记录，操作完成后释放锁。

### 3. 网络带宽资源的配额管理

系统在架构层面实施流量整形与配额控制。

**客户端自适应码率：**

系统不向所有用户发送统一的原始视频流。服务端预先生成低（480P）、中（720P）、高（1080P）三档码率的视频资源。

客户端播放器根据当前检测到的网络下行速度，动态请求对应档位的视频切片。

**API 速率限制：**

API 网关层实施全局限流策略。

**算法：**采用令牌桶算法（Token Bucket）。

**策略：**对每个 IP 或用户 ID 设置每秒最大请求数。超出配额的请求将直接被拒绝。

#### 4. 数据库连接与线程池管理

为了避免因资源耗尽导致的系统崩溃，架构强制要求对关键的系统级资源进行池化管理。

**数据库连接池：**

应用服务严禁在每次请求时临时创建和销毁数据库连接。必须使用 HikariCP 等高性能连接池组件，预先建立并维护一定数量的 TCP 连接。

**配置策略：**根据应用节点的 CPU 核心数设置最大连接数上限，防止过多的上下文切换降低性能。

**业务线程池隔离：**

采用 Hystrix 或 Sentinel 进行资源隔离。将“视频上传”、“推荐计算”等不同业务分配到独立的线程池中。

**目的：**当“视频上传”服务因故障响应缓慢并占满其线程池时，不会影响其他服务的正常资源获取，避免局部故障演变成全局雪崩。

#### 5. 全局配置中心

为了解决硬编码配置带来的维护难题，系统引入分布式配置中心作为全局共享资源。

**管理对象：**系统的动态参数，如“推荐算法的权重参数”、“敏感词库版本号”、“功能特性开关（Feature Flags）”。

**机制：**所有服务节点启动时从配置中心拉取最新配置，并保持长连接监听。当运营人员在控制台修改配置后，变更将毫秒级推送到所有服务器内存中生效，无需重启服务。

#### 5.9. 选择软件控制策略

为了确保系统在处理复杂的某些用户交互时保持响应迅速，同时在后台处理大规模视频数据时保持高效，我们分别为移动客户端和云端服务选择了不同的核心控制策略，并在特定业务流程中结合了状态机控制。

## 1. 移动客户端：事件驱动型控制策略

移动端 App 的核心任务是响应不可预测的用户输入（触摸、手势数据）。因此，客户端架构全面采用事件驱动型控制。

控制机制：

主事件循环：应用程序启动后进入一个无限循环，持续监听来自操作系统（Android）的事件分发。

事件处理器：将特定的业务逻辑绑定到具体的事件上。

具体应用场景：

用户界面交互：用户的点击（“点赞”）、滑动（“切换视频”）、长按（“录制视频”）被封装为标准事件对象。MVC 架构中的 Controller 充当事件监听器，接收这些事件并调用 Model 层的方法进行响应。

外部系统通知：网络状态变更（如从 Wi-Fi 切换到 4G）、系统低电量警告、推送通知到达等系统级广播，均通过广播接收器（Broadcast Receiver）以异步事件的方式触发应用逻辑。

优势：这种策略使得系统能够灵活地处理随机发生的用户行为，且不会因为等待某个操作（如等待用户点击）而阻塞 CPU 资源。

## 2. 云端 API 服务：过程驱动与并发控制结合

对于处理用户 HTTP 请求的应用服务器（如用户登录、获取视频列表），我们采用过程驱动型顺序控制，并嵌套在并发线程池模型中。

控制机制：

请求-响应模型：“调用-返回”（Call-Return）流程。当请求到达时，Web 容器接管控制权，按预定义顺序调用一系列对象方法：Controller -> Service -> DAO -> Database。

执行流：控制流在程序代码内部显式定义。

并发包装：

为了支持百万级用户，上述的过程驱动逻辑并非在单一主线程中运行。系统采用多

线程并发控制，为每一个到达的 HTTP 请求分配一个独立的线程。这些线程在逻辑上是并行的，互不干扰，由操作系统的调度器管理其物理执行时间片。

### 3. 视频后台处理：管道与过滤器控制策略

针对视频上传后的转码、审核与发布流程，这是一个线性的、数据转换型的任务，我们选择了管道控制策略。

控制机制：

我们将复杂的视频处理流程分解为一系列独立的、顺序执行的处理单元（过滤器）。前一个单元的输出数据流直接作为后一个单元的输入。

具体应用场景：

视频处理流水线：

阶段一 (Input) : 接收原始视频流。

阶段二 (Filter) : 格式校验与元数据提取。

阶段三 (Filter) : 病毒扫描。

阶段四 (Filter) : 自适应码率转码（生成 720P, 1080P）。

阶段五 (Filter) : 视频关键帧截取与封面生成。

阶段六 (Output) : 写入对象存储并分发至 CDN。

优势：这种策略使得复杂的处理逻辑变得清晰且易于维护。

### 4. 复杂交互逻辑：有限状态机控制

在客户端的“视频拍摄与编辑”这一核心功能中，我们采用有限状态机 (FSM) 来管理控制流。

控制机制：

定义一组明确的状态，以及在特定状态下允许触发的事件和转换规则。

具体应用场景：视频录制控制器

初始状态 (Idle): 仅响应“开始录制”事件，禁用“暂停”、“完成”按钮。

录制中状态 (Recording): 响应“暂停”、“停止”事件；系统周期性触发“写入帧”操作；若达到最大时长自动触发“完成”事件。

暂停状态 (Paused): 响应“继续录制”、“删除上一段”、“完成”事件。

**处理中状态 (Processing):** 禁用所有用户输入，显示加载动画，等待合成完成的回调事件。

**优势：**通过状态机控制，我们能够严密地封锁非法操作路径，防止因用户误操作（如快速连续点击按钮）导致的程序崩溃或数据损坏。

## 5. 异常与中断处理策略

**异常控制：**在上述所有策略中，均强制引入结构化异常处理（Try-Catch-Finally）。当发生不可预见的错误（如数据库连接断开）时，控制流立即跳转至异常处理模块，执行资源释放和错误日志记录，防止系统处于不一致状态。

**超时控制：**对于所有涉及网络通信或资源锁定的过程调用，均设置严格的超时中断机制。一旦操作超时，立即剥夺该线程的控制权并返回失败响应，防止单个卡死的任务耗尽全局系统资源。

## 5.10. 处理边界条件

为了确保“乐拍视界”在启动、退出或遭遇异常时表现出健壮性，系统架构明确定义了以下三类边界条件的处理策略：初始化（Initialization）、终止（Termination）以及故障（Failure）。

### 1. 初始化

系统组件从静止状态进入运行状态时，必须严格按照既定顺序加载资源和配置。

移动客户端启动流程：

**配置加载与完整性检查：**App 启动时，首先加载本地配置文件，并校验核心组件（如 ffmpeg 动态库）的完整性。若检测到文件损坏或版本不兼容，自动触发热修复补丁下载。

**权限申请与硬件预热：**在进入拍摄模块前，系统检查摄像头、麦克风及存储权限。若权限齐备，后台线程预初始化 Camera 硬件对象，以减少用户点击“拍摄”按钮时的等待延迟。

**网络探测与会话恢复：**系统自动检测当前网络环境（Wi-Fi/4G/无网）。同时，读取本地加密存储的 Token 尝试自动登录。若 Token 有效，立即建立 WebSocket 长连接以接收推送消息；若失效，则跳转至登录页。

**UI 骨架屏与缓存加载：**为避免启动白屏，优先从本地 SQLite 数据库加载上次缓存的首页视频列表和图片占位符，待网络请求返回最新数据后进行无缝替换。

**服务端服务启动流程：**

配置拉取：服务容器启动时，首先连接分布式配置中心，拉取最新的业务参数（如限流阈值、推荐算法权重）。

资源池预热：初始化数据库连接池和 Redis 连接池，建立最小空闲连接数，避免首批请求因建立 TCP 连接而超时。

服务注册：待所有内部组件初始化完毕且健康检查通过后，向服务注册中心注册自身 IP 和端口，正式对外接收流量。

## 2. 终止

当用户退出应用或服务器需要停机维护时，系统必须执行清理操作以释放资源并保存上下文。

**移动客户端终止：**

硬件资源释放：无论是用户主动杀掉进程还是系统因内存不足回收后台进程，Life-CycleObserver 均强制执行摄像头与音频驱动的释放操作，防止硬件被占用导致其他应用无法使用。

草稿自动保存：在拍摄或编辑过程中，若用户触发退出（或来电中断），系统捕获 onPause/onStop 事件，将当前的视频片段、编辑进度序列化写入本地文件系统，确保用户下次打开时能恢复现场。

后台任务管理：对于正在进行的视频上传任务，记录上传进度断点。应用退出时暂停上传，待下次启动时自动根据断点续传。

**服务端停机：**

流量截断：接收到停机信号（SIGTERM）时，服务首先从注册中心注销，停止接收新的外部请求。

在途请求处理：设置 N 秒的缓冲期，等待当前线程池中已接收的请求处理完毕。

数据落盘：强制将内存缓冲区中的日志、埋点数据、未提交的事务进行刷盘或提交，关闭数据库与消息队列连接，最后销毁进程。

## 3. 故障与失效

针对系统运行过程中不可避免的异常情况，设计了自动化的防御与恢复机制。

**网络异常边界：**

**弱网与断网：**当客户端检测到网络不可用时，UI 自动切换至“离线模式”，展示本地缓存的已浏览视频，并禁用点赞等在线交互功能。

**上传中断：**视频上传过程中若发生网络中断，系统自动触发指数退避（Exponential Backoff）重试机制。若多次重试失败，标记任务为“失败”，提示用户手动重试，而不是无限循环消耗电量。

**服务不可用边界（雪崩保护）：**

**熔断与降级：**当非核心服务（如个性化推荐引擎）响应超时或错误率超过阈值时，客户端自动触发降级策略，转为请求静态的“全站热门视频”列表，确保核心的视频播放功能不受影响。

**数据库故障：**若主数据库宕机，系统自动切换至只读模式，允许用户浏览视频，但暂时屏蔽发布、点赞等写入操作，并对用户进行友好提示。

**数据边界与异常值：**

**空状态处理（Cold Start）：**对于没有任何行为记录的新注册用户，推荐系统自动切换至“冷启动策略”，基于用户注册时选择的性别、年龄标签推送泛人群喜好的高热视频。

**超限防护：**对于超出系统处理能力的异常输入（如上传 10GB 的视频文件），API 网关层直接进行拦截并返回明确的错误码，防止后端服务崩溃。

## 5.11. 制订复用计划

为了在有限的开发周期内构建高性能的短视频社交平台，我们制定了严格的软件复用策略。复用计划分为三个层次：框架级复用（决定系统的控制流与基础架构）、库级复用（集成成熟的第三方功能组件）以及内部组件复用（提炼项目内的通用资产）。

### 1. 框架级复用

本项目核心架构基于 Qt 6 Framework 构建，利用其“C++ 后端 + QML 前端”的混合开发能力，实现跨平台的高性能与流畅交互。

**应用程序框架：**Qt Core & GUI

**复用策略：**直接利用 Qt 的事件循环作为主程序的控制中枢，实现控制反转。

**具体应用：**使用 QObject 对象模型提供的信号与槽机制，作为模块间通信的基础设施，替代传统的观察者模式手动实现，实现 UI 层（QML）与逻辑层（C++）的松耦合。

**网络框架：**复用 Qt Network 模块中的 QNetworkAccessManager 处理所有 HTTP/HTTPS 请求，复用其内置的线程池与异步回调机制，避免重复开发底层的 Socket

通信代码。

UI 渲染框架：Qt Quick (QML)

复用策略：利用 Qt Quick 的 Scene Graph 渲染引擎，该引擎底层直接调用 OpenGL/Vulkan/Metal。

具体应用：复用 QML 的声明式语法构建高动态的短视频滑动界面。利用 QAbstractListModel (C++) 与 ListView (QML) 的 Model-View-Delegate 架构，实现高性能的无限滚动列表，仅需编写特定的 Delegate (委托)。

## 2. 外部类库与组件复用

针对视频处理、数据库连接及特定算法需求，集成工业级的开源 C++ 库。

多媒体处理核心：FFmpeg

复用内容：编解码器 (libavcodec)、格式封装 (libavformat)、滤镜处理 (libavfilter)。

集成方式：编写 C++ 封装类 VideoEngine，将 FFmpeg 的 C 语言 API 封装为面向对象的 C++ 接口。复用其成熟的 H.265 解码能力和滤镜链 (Filter Graph) 功能，实现视频的剪辑、合并、水印添加及转码。

计算机视觉与特效：OpenCV

复用内容：图像处理算法库。

集成方式：用于客户端的“智能拍摄”模块。直接调用 OpenCV 的面部检测 (Face Detection) 和图像矩阵操作接口，实现美颜、磨皮及人脸关键点识别，以此为基础叠加动态贴纸。

数据库连接：libpqxx

复用内容：PostgreSQL 的官方 C++ 客户端库。

集成方式：服务端核心模块将复用 libpqxx。通过封装 DBConnectionPool 类，复用该库的连接管理能力。

## 3. 内部组件复用与构建

在开发过程中，我们将提炼具有高内聚性和通用性的代码模块，形成团队内部的资产库，供不同子系统或未来项目使用。

QML 通用 UI 组件库

定义： 将“乐拍视界”特有的视觉风格封装为独立的 QML 组件。

复用组件：

LePaiButton.qml：封装了统一的点击动效、圆角与渐变色风格。

VideoPlayerControl.qml：封装了播放、暂停、进度条拖拽逻辑的播放器外壳。

目标：确保全平台 UI 一致性，任何页面只需通过 import "components" 即可复用，无需重复编写样式代码。

C++ 核心逻辑库

网络请求封装器 (HttpClientWrapper)：

功能：基于 QNetworkAccessManager，封装了统一的 JWT Token 注入、请求签名 (Signature)、Gzip 解压缩及全局错误处理逻辑。

复用点：用户服务、视频服务、社交服务均通过此单例类发起请求。

通用工具类：

包括 TimeUtils（时间戳格式化）、CryptoUtils（MD5/SHA256 加密）、FileUtils（跨平台文件路径处理）。

#### 4. 设计模式复用

我们复用经过行业验证的通用设计模式来解决特定问题：

单例模式 (Singleton)：用于 ConfigurationManager（全局配置加载）和 SessionManager（用户登录状态持有），确保全局资源访问的唯一入口。

工厂模式 (Factory)：在视频处理模块中，使用工厂模式根据上传文件的扩展名 (.mp4, .mov, .avi) 创建对应的 Demuxer（解封装器）实例。

代理模式 (Proxy)：在图片加载中，使用代理模式显示低分辨率的缩略图占位，待高清图下载完成后再进行替换，优化滚动体验。

## 第6章详细设计

### 6.1. 类设计规范与标准

#### 1. 命名规范

类、属性与操作的命名约定（驼峰命名法等）

包与命名空间的映射规则（如 C++ 包结构）

#### 2. 可见性与访问控制

属性可见性设定（Private/Protected/Public）

操作可见性设定与封装边界

#### 3. 数据类型标准

基础数据类型与目标编程语言（C++）类型的映射

复杂数据类型与值对象（Value Object）的定义

### 6.2. 类的详细设计与精化

#### 1. 实体类精化

核心业务实体（用户、视频、音乐）

属性的详细定义（名称、类型、初始值、多重性）

派生特性的处理（如：视频的“实时热度”计算）

完整性约束的定义（如：视频时长限制、格式校验）

辅助实体与值对象

视频元数据对象

用户权限凭证对象

#### 2. 控制类设计

客户端控制器（MVC Controller）

拍摄控制器（ShootController）：处理摄像头信号与用户输入

播放控制器（VideoPlayerController）：处理流媒体缓冲与手势交互

服务端业务逻辑控制器

视频发布服务（VideoPublishService）：协调上传、转码、持久化

**推荐计算服务 (RecommendationService) : 协调画像与召回算法**

### 3. 边界类设计

移动端视图组件

沉浸式播放容器类

视频拍摄与编辑界面类

服务端接口类

API 接口定义类

外部服务适配器类 (CDN 适配器)

### 6.3. 操作与方法设计

#### 1. 操作签名定义

核心方法的参数列表 (名称、类型、顺序)

返回值类型与异常处理机制

操作的前置条件与后置条件

#### 2. 对象生命周期与状态转换

视频对象状态机设计

上传中 -> 转码中 -> 审核中 -> 已发布/被驳回的状态流转逻辑

播放器状态机设计

空闲 -> 加载中 -> 播放中 -> 暂停 -> 缓冲的状态控制逻辑

#### 3. 关键算法逻辑

视频节拍自动对齐算法逻辑 (Beat Matching)

视频推荐加权算法逻辑 (基于用户行为的实时权重计算)

缓存淘汰算法逻辑 (移动端 LRU 策略实现)

### 6.4. 关联与结构设计

#### 1. 类之间的关联实现

一对多关联的实现策略 (如：用户与视频列表，使用集合类容器)

多对多关联的拆解 (如：用户与关注用户的关系表设计)

关联的导航方向（单向/双向）与引用实现

## 2. 依赖管理与解耦

接口定义与实现类的分离

观察者模式在 MVC 组件交互中的具体实现（Event/Listener 机制）

代理模式在网络请求与图片加载中的应用设计

## 6.5. 数据存储详细设计

### 1. 关系型数据库模式

表结构定义（User, Video, Comment, Relation 表）

字段类型、长度、约束（主键、外键、非空）

索引设计优化（针对查询热点）

### 2. 非结构化存储设计

对象存储（OSS）的目录结构与命名规则（Hash 打散策略）

CDN 缓存策略配置

### 3. 缓存数据结构（Redis）

Key-Value 命名规范

计数器（点赞、播放）的原子操作设计

热门视频列表的 ZSet 设计

## 6.6. 接口详细说明

### 1. 客户端与服务端交互接口

视频流获取接口（参数、响应格式 JSON/Protobuf）

互动操作接口（点赞、关注）

文件上传接口（分片上传协议）

### 2. 内部子系统接口

推荐引擎调用接口

视频转码服务回调接口

## 后记

正文内容，方正仿宋，小四，首行缩进。正文内容，方正仿宋，小四，首行缩进。正文内容，方正仿宋，小四，首行缩进。

## 参考文献

- [1] YOUNG.RSS 是什么? [EB/OL]. <http://jingpin.org/what-is-rss/>.
- [2] 杨博, 彭博.RSS 提要分析与阅读器设计[R].成都: 四川大学计算机学院, 2007: 42-43.
- [3] 逸出络然.RSS 技术的原理[EB/OL].<http://yclran.blog.163.com/blog/static/979454962009111034111558/>.
- [4] 佚名.Qt 是什么[EB/OL]. <http://qt.nokia.com/title-cn>.
- [5] 佚名.Model/View Programming[EB/OL]. <http://doc.trolltech.com/4.6/model-view-programming.html>.
- [6] [加拿大]Jasmin Blanchette[英]Mark Summerfield 著 闫锋欣,曾泉人,张志强译.
- [7] C++ GUI Qt4 编程 (第二版) [M].电子工业出版社: 2008:182-206,291-305.
- [8] 佚名.XML Processing[EB/OL]. <http://doc.trolltech.com/4.6/xml-processing.html>.
- [9] Michael Blala James Rumbangh 著.UML 面向对象建模与设计 (第 2 版) [M].北京: 人民邮电出版社,2006:136-235.
- [10]胡海静,王育平,等. XML 技术精粹[M]. 北京: 机械工业出版社, 2001:17-19.