**需求文档**

**一、项目概述**

**1.1 项目背景**

随着互联网广告行业的快速发展，企业对广告投放效率和用户体验的要求日益提高。本项目旨在构建一个基于Web的互联网广告平台，采用现代化的微服务架构（Spring Boot + Spring Cloud），为企业提供广告购买、审核、投放、统计分析及财务管理等一站式解决方案。

**1.2 项目目标**

实现广告主与媒体方的高效对接，支持广告的全生命周期管理。

提供广告投放效果监控与数据分析能力，优化广告精准度。

支持高并发场景下的稳定运行，保障广告请求的低延迟响应。

通过微服务架构实现系统的模块化、可扩展性和高可用性。

**二、场景分析：**

**场景1：广告购买与审核流程**

1. 用户提交广告订单 → 选择广告类型、预算、投放时间。

2. 系统检查账户余额是否充足 → 不足则引导充值。

3. 广告内容进入审核队列 → 自动审核（敏感词/图片识别） + 人工复审。

4. 审核通过 → 扣除账户余额，广告上线；不通过 → 通知用户修改。

**场景2：广告充值（模拟支付）**

1. 用户选择充值金额 → 跳转模拟支付页面。

2. 模拟支付成功 → 更新用户账户余额，生成充值记录。

3. 充值记录可关联至广告订单或发票申请。

**场景3：广告展示与统计**

1. 前端调用广告接口 → 携带用户特征（地理位置、设备类型）。

2. 后端根据策略选择广告 → 记录展示日志（时间、曝光量）。

3. 用户点击广告 → 更新点击量，实时统计效果。

**场景4：发票开具**

1. 用户申请发票 → 关联充值记录或广告订单。

2. 财务系统生成PDF发票 → 更新开票状态，通知用户下载。

3. 支持电子发票（API对接）或手动上传。

**三、基于Web的互联网广告平台异常情况分析:**

**1. 广告审核异常**

* **广告内容违规**：广告客户提交的广告内容可能包含违法违规信息，如虚假宣传、低俗内容等。平台需在审核阶段严格把控，一旦发现违规内容，应立即驳回并通知广告客户修改。
* **审核超时**：由于广告提交量过大或审核人员不足，可能导致广告审核时间过长，影响广告客户的投放计划。平台可以通过设置审核优先级、增加审核人员或采用自动化审核工具来缓解这一问题。
* **审核标准不一致**：不同审核人员对广告内容的判断可能存在差异，导致审核结果不一致。平台应制定明确的审核标准和流程，定期对审核人员进行培训，确保审核的公正性和一致性。

**2. 支付异常**

* **支付失败**：用户在充值过程中可能遇到支付失败的情况，如网络问题、支付平台故障、账户余额不足等。平台应及时捕获支付失败的异常，并向用户提示具体原因，同时提供重试或更换支付方式的选项。
* **重复支付**：由于网络延迟或用户误操作，可能导致用户重复提交支付请求。平台需要在后端进行支付状态的校验，避免重复扣款，并及时向用户反馈支付状态。
* **支付接口异常**：第三方支付接口可能出现故障或维护，导致支付无法正常进行。平台应提前与支付服务提供商沟通，了解接口的维护计划，并在系统中设置备用支付方案，以减少对用户的影响。

**3. 数据异常**

* **数据丢失或损坏**：由于服务器故障、数据库损坏或人为误操作，可能导致用户数据、广告数据或充值记录丢失或损坏。平台应定期备份数据，并制定数据恢复策略，确保在数据丢失时能够快速恢复。
* **数据查询异常**：用户在查询广告展示情况或充值历史时，可能遇到查询结果不准确或查询超时的情况。平台需要优化数据库查询性能，合理设计索引，并对查询结果进行缓存，提高查询效率。
* **数据统计错误**：广告展示次数、点击次数或转化率等数据统计可能出现错误，导致广告客户对平台数据产生质疑。平台应定期校验数据统计逻辑，确保数据的准确性和可靠性。

**4. 用户操作异常**

* **账户被盗**：用户的账户信息可能被泄露，导致账户被盗用。平台应提供账户安全保护措施，如密码加密、二次验证等，并在用户登录时进行身份验证，确保账户安全。
* **用户误操作**：用户可能在操作过程中出现误操作，如误删除广告或充值错误金额。平台应在操作前提供明确的提示信息，并在操作后提供撤销或修改选项，减少用户损失。

**5. 系统性能异常**

随着用户量和广告量的增加，平台可能出现以下性能异常情况：

* **服务器负载过高**：在高峰时段，服务器可能因负载过高导致响应缓慢或服务不可用。平台可以通过增加服务器资源、优化代码性能、采用负载均衡技术等方式来缓解这一问题。
* **页面加载缓慢**：前端页面可能因资源过多或代码优化不足导致加载缓慢。平台应优化前端代码，减少不必要的资源加载，并采用CDN加速技术，提高页面加载速度。
* **并发访问异常**：在高并发情况下，平台可能出现数据库连接池耗尽、线程池阻塞等问题。平台需要合理配置资源池大小，并通过分布式架构和缓存技术来提高系统的并发处理能力。

**四、需求分析**

**4.1功能需求**

**核心功能模块**

模块名称 功能描述

广告购买与审核 广告主提交广告创意，管理员审核并发布广告。

广告充值（模拟支付） 广告主账户充值，模拟支付流程，记录交易记录。

充值历史查询 广告主可查看账户的充值记录及余额明细。

广告管理 广告主管理已购买的广告，包括修改、暂停、删除等操作。

广告展示与查询 用户端展示广告，广告主可查询广告的曝光量、点击率等数据。

前台广告接口 提供标准化的API接口，供第三方媒体调用广告资源。

发票开具 根据广告投放费用生成电子发票，支持下载与打印。

**扩展功能**

多租户支持：不同广告主共享同一系统，但数据隔离。

实时竞价（RTB）：支持程序化广告投放，动态调整广告位价格。

用户画像分析：基于行为数据生成用户标签，优化广告匹配。

**4.2 非功能需求**

类型 要求

性能 单节点支持每秒1000+广告请求，响应时间≤200ms。

可靠性 系统可用性≥99.9%，支持故障自动恢复。

安全性 数据加密传输（HTTPS），用户身份认证（OAuth2），防止SQL注入和XSS攻击。

扩展性 微服务可独立扩展，支持水平扩容应对流量高峰。

易用性 提供图形化管理后台，简化广告主的操作流程。

**4.3 技术架构**

整体架构图

整体架构图

┌───────────────┐ ┌───────────────┐ ┌───────────────┐

│ Advert-Gateway │ → │ Advert-Eureka │ ← │ Advert-Service │

└───────────────┘ └───────────────┘ └───────────────┘

▲ ▲ ▲

│ │ │

┌───────────────┐ ┌───────────────┐ ┌───────────────┐

│ Advert-API │ │ Advert-Admin │ │ Advert-Pay │

└───────────────┘ └───────────────┘ └───────────────┘

Advert-Gateway：API网关，统一管理路由、鉴权和限流。

Advert-Eureka：服务注册中心，实现微服务的自动发现与负载均衡。

Advert-Service：核心业务模块，包含广告管理、用户管理、支付管理等微服务。

技术选型

组件 技术/工具

框架 Spring Boot 3.x, Spring Cloud 2023.x

数据库 MySQL 8.0（关系型数据），Redis（缓存），Elasticsearch（日志与搜索）

消息队列 RabbitMQ/Kafka（异步处理广告请求与支付回调）

分布式配置 Spring Cloud Config + Git（集中管理配置文件）

安全框架 Spring Security + OAuth2（JWT令牌认证）

监控与日志 Prometheus（指标采集），Grafana（可视化），ELK（日志分析）

容器化部署 Docker + Kubernetes（容器编排与自动化部署）

**4.4 模块划分与实现**

1. 广告管理模块（Advert-Service）

功能：广告创意管理、投放策略设置、广告审核流程。

实现方式：

使用@RestController暴露REST API，通过FeignClient调用其他服务。

广告状态机（如“待审核”→“已发布”→“已结束”）通过状态枚举类实现。

广告审核逻辑封装为独立服务，支持人工审核与自动审核（基于规则引擎）。

2. 支付管理模块（Advert-Pay）

功能：模拟支付接口、充值记录持久化、余额计算。

实现方式：

使用MockPaymentService模拟第三方支付接口（如支付宝、微信）。

Redis缓存用户余额，减少数据库高频访问。

支付结果通过消息队列异步通知下游服务（如广告投放模块）。

3. 用户管理模块（Advert-User）

功能：用户注册/登录、角色权限管理（广告主、管理员）。

实现方式：

JWT令牌生成与验证（通过Spring Security Opaque Token模式）。

角色权限通过数据库表关联（user\_role表定义用户与角色的关系）。

4. 广告投放模块（Advert-Delivery）

功能：根据用户画像匹配广告，实时返回广告素材。

实现方式：

使用Redis缓存热门广告资源，降低数据库压力。

基于Elasticsearch的广告索引，支持多条件快速检索（如地域、设备类型）。

广告调度算法：轮询、加权随机、ECPM（千次曝光收益最大化）。

5. 数据统计模块（Advert-Analytics）

功能：统计广告曝光量、点击率、转化率等关键指标。

实现方式：

日志埋点：通过AOP切面记录广告请求与用户行为（点击、浏览）。

数据聚合：定时任务将原始日志写入MySQL，生成汇总报表。

实时看板：通过WebSocket推送统计数据至前端（ECharts图表）。

6. 发票管理模块（Advert-Invoice）

功能：生成电子发票并提供下载链接。

实现方式：

使用iText库生成PDF格式发票，存储至MinIO对象存储。

发票状态跟踪（已申请→已生成→已下载）。

**4.5 数据库设计**

**1. 核心表结构设计**

**1.1 用户表（User）**

用途：存储广告主、管理员等用户的基本信息及账户状态。

字段设计：

sql

CREATE TABLE user (

id BIGINT PRIMARY KEY AUTO\_INCREMENT, -- 用户唯一标识（主键）

username VARCHAR(50) UNIQUE NOT NULL, -- 用户名（唯一且非空）

password VARCHAR(100) NOT NULL, -- 密码（加密存储）

role ENUM('ADVERTISER', 'ADMIN') NOT NULL, -- 角色（广告主/管理员）

balance DECIMAL(10,2) DEFAULT 0.00, -- 账户余额（默认值为0）

created\_at DATETIME NOT NULL, -- 注册时间

updated\_at DATETIME NOT NULL -- 最后更新时间

);

关键点：

主键约束：id作为主键，唯一标识每条用户记录。

唯一约束：username字段唯一，避免重复注册。

角色枚举：通过ENUM限制角色范围，简化权限管理。

索引优化：对频繁查询的字段（如username）创建索引。

**1.2 广告表（Ad）**

用途：存储广告创意、投放规则及状态。

字段设计：

sql

CREATE TABLE ad (

id BIGINT PRIMARY KEY AUTO\_INCREMENT, -- 广告唯一标识

title VARCHAR(255) NOT NULL, -- 广告标题

content TEXT, -- 广告内容（支持富文本）

image\_url VARCHAR(255), -- 图片地址

link\_url VARCHAR(255) NOT NULL, -- 点击跳转链接

status ENUM('PENDING', 'ACTIVE', 'EXPIRED') NOT NULL, -- 广告状态

start\_time DATETIME, -- 投放开始时间

end\_time DATETIME, -- 投放结束时间

user\_id BIGINT NOT NULL, -- 关联广告主ID（外键）

create\_time DATETIME NOT NULL, -- 创建时间

update\_time DATETIME NOT NULL, -- 更新时间

FOREIGN KEY (user\_id) REFERENCES user(id) -- 外键约束

);

关键点：

外键约束：user\_id关联到user表，确保广告归属明确。

状态机设计：status字段定义广告生命周期（待审核→已发布→已过期）。

索引优化：对user\_id和status创建索引，加速条件查询。

**1.3 充值记录表（Recharge）**

用途：记录广告主的账户充值明细。

字段设计：

sql

CREATE TABLE recharge (

id BIGINT PRIMARY KEY AUTO\_INCREMENT, -- 充值记录ID

user\_id BIGINT NOT NULL, -- 关联广告主ID

amount DECIMAL(10,2) NOT NULL, -- 充值金额

transaction\_id VARCHAR(50) UNIQUE NOT NULL, -- 第三方支付流水号

payment\_method VARCHAR(50) NOT NULL, -- 支付方式（如支付宝、微信）

status ENUM('SUCCESS', 'FAILED') NOT NULL, -- 支付状态

created\_at DATETIME NOT NULL, -- 充值时间

FOREIGN KEY (user\_id) REFERENCES user(id) -- 外键约束

);

关键点：

事务一致性：通过外键和事务机制保证充值与账户余额同步。

幂等性设计：transaction\_id唯一约束，防止重复充值。

**1.4 广告投放记录表（Ad\_Impression）**

用途：记录广告曝光和点击行为。

字段设计：

sql

CREATE TABLE ad\_impression (

id BIGINT PRIMARY KEY AUTO\_INCREMENT, -- 记录ID

ad\_id BIGINT NOT NULL, -- 关联广告ID

user\_id BIGINT NOT NULL, -- 关联用户ID（可选）

impression\_time DATETIME NOT NULL, -- 曝光时间

clicked BOOLEAN DEFAULT FALSE, -- 是否点击

ip\_address VARCHAR(45), -- 用户IP地址

device\_type VARCHAR(50), -- 设备类型（PC/移动端）

FOREIGN KEY (ad\_id) REFERENCES ad(id) -- 外键约束

);

关键点：

高并发场景：采用列式存储（如Hologres）或分区表优化大数据量写入。

统计分析：通过聚合impression\_time和clicked字段计算曝光率、点击率。

1.5 发票表（Invoice）

用途：存储广告费用发票信息。

字段设计：

sql

CREATE TABLE invoice (

id BIGINT PRIMARY KEY AUTO\_INCREMENT, -- 发票ID

user\_id BIGINT NOT NULL, -- 关联广告主ID

total\_amount DECIMAL(10,2) NOT NULL, -- 总金额

invoice\_number VARCHAR(50) UNIQUE NOT NULL, -- 发票编号

issue\_date DATETIME NOT NULL, -- 开票日期

status ENUM('PENDING', 'ISSUED', 'DOWNLOADED'),-- 发票状态

pdf\_url VARCHAR(255), -- PDF文件存储地址

FOREIGN KEY (user\_id) REFERENCES user(id) -- 外键约束

);

**五、开发与部署计划**

**5.1 开发阶段**

阶段 时间范围 主要任务

需求分析 第1周 明确功能边界，输出PRD文档。

技术选型与原型设计 第2周 确定微服务架构，搭建Spring Cloud基础环境。

核心模块开发 第3-5周 实现广告管理、支付管理、用户管理三大核心模块。

集成与测试 第6周 接入消息队列、数据库优化，编写单元测试与集成测试用例。

上线准备 第7周 部署到Kubernetes集群，配置监控与报警规则。

**5.2 部署方案**

容器化部署：使用Docker镜像打包各微服务，通过Kubernetes进行容器编排。

CI/CD流水线：Jenkins/GitLab CI自动触发构建、测试与部署流程。

灰度发布：通过Istio实现金丝雀发布，逐步验证新版本稳定性。