**大模型代理平台软件设计文档**

**1. 引言**

**1.1 项目背景**

随着大语言模型（LLM）和生成式AI技术的飞速发展，开发者和企业常常需要集成和使用来自不同提供商的多种模型。每个模型都有其独特的API接口、能力标签、认证方式和数据格式，这导致了集成成本高、管理复杂、切换困难等问题。

**1.2 设计目标**

本项目旨在设计并实现一个**大模型代理后端（LLM-Proxy）**。其核心目标是提供一个统一、稳定且可管理的API网关，将多样化的上游AI模型能力进行封装和抽象，为最终用户或客户端应用提供一个简洁、一致的调用接口。

主要设计目标包括：

* **统一接口**: 为对话、文生图等核心AI能力提供统一的API，屏蔽底层模型差异。
* **集中管理**: 提供管理后台功能，用于动态配置、监控和管理所有接入的模型。
* **安全认证**: 建立两层认证体系，区分管理员和终端用户，确保系统和API调用的安全。
* **智能路由**: 实现基于模型能力和优先级的智能路由策略，提高系统的灵活性和健壮性。

**2. 系统概述**

本系统在架构上可分为三个核心部分，它们协同工作，共同构成了整个代理服务。

1. **管理员 (Admin)**: 系统的维护者。通过Web界面或直接调用API，对系统进行配置，包括注册登录、管理模型池、生成和管理供外部使用的访问密钥（Access Key）。
2. **终端用户/应用 (End-User/Application)**: AI能力的消费者。使用访问密钥，通过调用代理服务提供的统一AI接口来获取对话、图片生成等服务。
3. **代理后端 (Proxy Backend)**: 系统的核心。它负责处理所有来自管理员和终端用户的请求。其内部逻辑包括：
   * 验证请求的合法性（管理员的JWT Token或用户的Access Key）。
   * 根据请求内容，从已配置的模型池中选择一个合适的上游模型。
   * 将用户的通用请求格式转换为特定上游模型所需的格式。
   * 调用上游模型API，并将返回结果转换为统一的响应格式，再返回给用户。

**3. 功能模块划分**

根据系统职责，我们将后端系统划分为三大核心功能模块。

**3.1 认证与授权模块 (Authentication)**

* **职责**: 负责用户的注册、登录、鉴权以及access key的创建和管理。
* **说明**: 这是系统的安全基石。它实现了两级认证体系：
  1. **管理员认证**: 基于用户名/密码，通过JWT（JSON Web Token）实现。管理员登录后获取token，用于访问模型管理等高权限接口。
  2. **终端用户认证**: 基于Access Key。管理员为应用或用户创建Access Key，终端用户在调用AI能力接口时，需在请求头中携带此Key。

**3.2 模型管理模块 (Model Management)**

* **职责**: 负责对所有接入的第三方大模型进行生命周期管理。
* **说明**: 该模块为管理员提供了一套完整的CRUD（创建、读取、更新、删除）接口，用于维护一个动态的模型池。每个模型都作为一条配置记录存储，包含其API地址、密钥、能力（如text-to-text, text-to-image）和**优先级**等关键信息。这个模块是实现智能路由和模型热切换的基础。

**3.3 AI能力代理模块 (AI Capability Proxy)**

* **职责**: 负责接收终端用户的AI服务请求，并将其代理到合适的上游模型。
* **说明**: 这是系统的核心业务模块，也是对外的价值所在。它定义了一套标准化的AI服务接口，如对话（Chat）和图片生成（Image Generation）。该模块内部包含了**请求转换**、**智能路由**和**响应封装**的复杂逻辑，对用户完全透明。

**4. 模块详细设计**

**4.1 认证与授权模块**

**4.1.1 核心设计**

本模块的设计核心是**职责分离**。管理员的身份管理与服务的访问凭证是分开的。管理员通过JWT来证明“我是管理员”，而终端用户通过Access Key来证明“我有权使用服务”。

**4.1.2 主要接口设计**

* **管理员注册 (POST /v1/auth/register)**: 提供一个初始创建管理员账户的入口。
* **管理员登录 (POST /v1/auth/login)**: 管理员使用用户名和密码进行登录，成功后系统会签发一个具有时效性的JWT Token。此token是后续所有管理操作的凭证。
* **Access Key管理**:
  + **创建 (POST /v1/auth/access-keys)**: 已登录的管理员调用此接口，为终端用户或应用生成一个新的Access Key。这个Key是无状态的，代理后端只需在数据库中验证其有效性即可。
  + **查询 (GET /v1/auth/access-keys)**: 管理员可以查看自己创建的所有Access Key及其状态，便于审计和管理。
  + **删除 (DELETE /v1/auth/access-keys/{id})**: 管理员可以随时吊销某个Access Key的访问权限。

**4.2 模型管理模块**

**4.2.1 核心设计**

本模块的核心是**模型配置的抽象化**。无论底层是OpenAI的GPT-4还是DALL-E 3，在本系统中都被抽象为一个具有标准属性的Model对象。关键属性包括：

* modelIdentifier: 模型的唯一标识符，用于用户指定模型。
* capabilities: 描述模型能做什么的标签数组（例如 ["text-to-text", "text-to-image"]），这是智能路由的关键依据。
* priority: 一个整数，数字越小，优先级越高。当多个模型满足同一能力时，系统会优先选择优先级最高的。
* status: 模型的在线/离线状态，允许管理员一键启用或禁用某个模型，而不必删除其配置。

**4.2.2 主要接口设计**

* **创建模型 (POST /v1/models)**: 管理员通过此接口录入一个新的模型配置，包括其名称、标识、API密钥、能力和优先级等。
* **获取模型列表 (GET /v1/models)**: 提供强大的分页、筛选（按能力、状态）和排序（按优先级、创建时间）功能，方便管理员在UI中展示和管理模型。
* **更新模型 (PUT /v1/models/{id})**: 允许管理员修改已存在模型的配置信息。
* **更新模型状态 (POST /v1/models/{id}/status)**: 提供一个轻量级的接口，专门用于快速切换模型的上线/下线状态，操作更便捷。
* **删除模型 (DELETE /v1/models/{id})**: 从系统中彻底移除一个模型配置。

**4.3 AI能力代理模块**

**4.3.1 核心设计**

本模块的设计核心是**请求的抽象与智能分发**。

1. **统一入口**: 为一类AI能力（如所有类型的对话）提供单一的API入口。例如，POST /v1/chat 接口通过检查请求体中是否存在images字段，来智能地区分处理**纯文本对话**和**多模态视觉对话**。这种设计极大地简化了客户端的调用逻辑。
2. **智能路由逻辑**:
   * **精确匹配**: 如果用户在请求中明确指定了modelIdentifier或modelInternalId，系统将直接使用该模型。
   * **自动选择**: 如果用户未指定模型，系统将： a. 根据请求类型确定所需的能力（例如，调用/v1/chat且带有images字段，则需要vision能力）。 b. 从模型池中筛选出所有具备该能力且状态为“上线”的模型。 c. 在筛选出的模型中，选择priority数值最小（优先级最高）的模型来执行任务。
3. **适配与转换**: 在确定了目标模型后，代理模块会将统一的请求体转换为目标模型API所要求的特定格式，并在收到响应后，再将其转换为统一的响应体返回给用户。

**4.3.2 主要接口设计**

* **对话接口 (POST /v1/chat)**:
  + **功能**: 作为所有对话类请求的统一入口。
  + **设计亮点**: 通过images字段的有无，在内部自动切换**文生文**和**图文对话**两种模式，无需提供两个独立的API。支持多轮对话历史（history）和丰富的模型参数（options）。
* **图片生成接口 (POST /v1/generate-image)**:
  + **功能**: 作为所有图片生成和编辑类请求的统一入口。
  + **设计亮点**: 通过originImage字段的有无，在内部自动切换**文生图**（Text-to-Image）和**图生图/编辑**（Image-to-Image）两种模式。

**5. 总结**

本系统通过**认证授权**、**模型管理**和**AI能力代理**三大模块的协同工作，系统能够有效地解决多模型集成带来的复杂性和管理难题。其核心优势在于**接口的统一性**、**管理的灵活性**和**路由的智能化**，为上层应用提供了一个稳定、高效且易于扩展的AI能力调用平台。