# API 设计与集成（Python + FastAPI + Celery）

目标：提供清晰稳定的后端接口，支持上传、解析、个性化重写、素材生成与结果查询；采用 Celery 异步化耗时步骤；以 LLMProvider 抽象统一多家模型 API；用 Pydantic 保证结构化输入/输出。

## 技术选型要点

* 模型使用策略：MODEL\_SERVE\_MODE=api（仅云端 API，不做本地训练/部署）
* 框架：FastAPI（类型提示 + OpenAPI 自动化）
* 队列：Celery + Redis/RabbitMQ（ingest\_pdf、personalize\_text、generate\_materials、score\_eval）
* LLM：官方/厂商 SDK，通过 LLMProvider 封装（complete/chat/embed）
* 结构化输出：Pydantic/JSON Schema；必要时“守护提示”+ 二次校对
* 网络/重试：httpx + tenacity/backoff（超时、重试、退避）

## LLMProvider 接口（示例）

class LLMProvider(Protocol):
async def complete(self, prompt: str, \*\*kwargs) -> str: ...
async def chat(self, messages: list[dict], \*\*kwargs) -> str: ...
async def embed(self, texts: list[str]) -> list[list[float]]: ...

* 通过环境变量切换厂商/模型：LLM\_PROVIDER、LLM\_MODEL、EMBEDDING\_MODEL。
* 对每次调用记录：提示词版本、模型、延迟、token 用量，用于 A/B 与回滚。

## 核心端点与异步编排

* POST /ingest/pdf：上传 PDF → 投递 ingest\_pdf 任务，返回 taskId
* GET /ingest/tasks/:id：查询解析进度与结果
* POST /personalize：提交 profile + chunks → 投递 personalize\_text，返回 taskId
* POST /materials：提交个性化文本 → 投递 generate\_materials，返回 taskId
* GET /materials/:id：查询素材生成结果

状态机（Celery Task State）：PENDING -> STARTED -> RETRY -> FAILURE | SUCCESS，必要时维护业务级 progress(0-100) + stage。

## 响应格式与错误码

* 统一格式：{ code, message, data }，成功 code=0
* 常见错误：
* 1001 参数校验失败（Pydantic）
* 2001 上游模型超时/限流（自动重试后仍失败）
* 3001 存储/向量检索异常
* 4001 未授权/鉴权失败

## 结构化输出与校验

* 让 LLM 输出 JSON（或 JSON Lines），后端以 Pydantic 校验：

class QuizQuestion(BaseModel):
id: str
type: Literal["single","multi","tf","short"]
stem: str
options: list[str] | None
answer: Any
explanation: str

* 校验失败即视为模型错误并触发重试/降级；可用 jsonschema/zod（前端）双向校验。

## 契约测试与 E2E

* OpenAPI：自动生成规范，使用 schemathesis/pact 做契约回归
* E2E：curl/Postman 覆盖关键链路（上传→解析→个性化→素材）
* 负载：k6 模拟限流/退避路径

## 观测与追踪

* 结构化日志（JSON），关键字段：taskId、profileId、promptVersion、model、latency
* 指标：任务成功率、P95 延迟、重试次数、评测均分
* 追踪：OpenTelemetry（FastAPI/Celery 链路）

## 安全与限流

* 鉴权：API Key / OAuth（按产品形态选择）
* 速率限制：Nginx/API Gateway + 应用侧 backoff/令牌桶
* 数据：敏感信息不下发到客户端；PDF/解析结果按租户隔离