# Freepik API 中转服务系统开发方案

本文档设计一个基于 **Next.js + Supabase + Cloudflare R2** 的中转服务系统（B），用于统一代理管理多个 C 站点对 Freepik 平台的图像生成、视频生成和图像编辑 API 的调用。系统应支持任务接收转发、API Key 轮询、Webhook 处理、调用统计和限流，以及内部监控界面。以下分模块详细说明设计思路。

## 接口处理逻辑设计

* **任务接收与入库**：B 系统对外暴露 RESTful 接口（如 /api/task），供 C 站点提交任务请求。请求中包含业务参数（如 prompt、参考图 URL、样式 ID、分辨率等）和 C 站点自定义的回调地址（callback\_url）。收到请求后，B 在 Supabase 数据库的 tasks 表中创建一条记录，状态标记为 PENDING，并记录请求参数及所属 C 站点信息。
* **API Key 选择**：B 从数据库 api\_keys 表轮询可用的 Freepik API Key，确保所选 Key 当日调用量未达上限[[1]](https://docs.freepik.com/ratelimits#:~:text=,Rate%20per%20seconds)。可采用循环队列或随机轮询的方式，记录选用的 Key 和更新 used\_today 计数。
* **请求转发**：根据任务类型调用对应 Freepik API：
* **图像生成**：调用 Freepik 图像生成端点（如 Mystic、Classic Fast 等），URL 形如 POST https://api.freepik.com/v1/ai/{model}。请求体采用 JSON，至少包括 prompt、webhook\_url（回调至 B 的地址），并根据模型附加参数（如 structure\_reference、style\_reference、resolution、aspect\_ratio 等）[[2]](https://docs.freepik.com/api-reference/mystic/post-mystic#:~:text=,aSDinaTvuI8gbWludGxpZnk)。例如 Mystic 模式示例请求：
* {  
   "prompt": "描述性文本",  
   "webhook\_url": "https://b.example.com/api/webhook/freepik",  
   "structure\_reference": "...", "structure\_strength": 50,  
   "style\_reference": "...",  
   "resolution": "2k", "aspect\_ratio": "square\_1\_1",  
   "model": "realism", "creative\_detailing": 33,  
   "filter\_nsfw": true, "styling": { ... }  
  }
* Freepik 返回状态 200 及 JSON 包含 data.task\_id（任务 ID）和 data.status（如 "IN\_PROGRESS")[[3]](https://docs.freepik.com/api-reference/mystic/post-mystic#:~:text=,6489e6daee91%22%2C%20%22status%22%3A%20%22IN_PROGRESS%22)。B 将任务状态更新为 IN\_PROGRESS，并记录 Freepik 返回的 task\_id 和所用 API Key。
* **视频生成**：调用 Freepik 视频生成端点，如 POST https://api.freepik.com/v1/ai/image-to-video/{model}。示例参数包括 prompt、first\_frame\_image（图片 URL）、duration（秒数）、prompt\_optimizer（布尔）等[[4]](https://docs.freepik.com/api-reference/image-to-video/minimax-hailuo-02-768p/post-minimax-hailuo-02-768p#:~:text=,7890.jpg%22%2C%20%22duration%22%3A%206)。例如 MiniMax 模式示例：
* {  
   "prompt": "山间日落，鸟儿飞翔",  
   "first\_frame\_image": "https://.../sunset.jpg",  
   "duration": 6, "prompt\_optimizer": true,  
   "webhook\_url": "https://b.example.com/api/webhook/freepik"  
  }
* 返回示例同样包含 "task\_id" 和 "status": "IN\_PROGRESS"[[5]](https://docs.freepik.com/api-reference/image-to-video/minimax-hailuo-02-768p/post-minimax-hailuo-02-768p#:~:text=%7B%20%22data%22%3A%20%7B%20%22task_id%22%3A%20%22046b6c7f,)。
* **图像编辑**：根据不同功能调用不同端点。例如去背景接口为 POST /v1/ai/beta/remove-background，要求表单格式并提供 image\_url 参数[[6]](https://docs.freepik.com/api-reference/remove-background/post-beta-remove-background#:~:text=)；放大（Upscaler）接口为异步端点 /v1/ai/image-upscaler（返回 task\_id）；风格迁移、重新光照等则调用各自端点。B 需将 C 站点传入的原图 URL 或参数转发给 Freepik。同步接口（如去背景）直接返回结果，异步接口返回 task\_id。
* **异步任务处理**：对返回 IN\_PROGRESS 的调用，B 将任务状态保持为 IN\_PROGRESS。此后有两种方式获取结果：
* **Webhook 回调**：Freepik 完成任务后会向 B 指定的 webhook\_url 发出回调，回调内容为 JSON，通常包含 data.task\_id、data.status（COMPLETED 或 FAILED）、以及生成内容链接等。B 对此回调的 /api/webhook/freepik 路径进行处理，找到对应任务（匹配 task\_id），更新状态，并记录回调中可能返回的结果列表 URL。
* **轮询查询**：若回调失败或延迟，B 可以使用后台计划任务（如定时 CRON 或队列）定期调用 Freepik 的状态查询接口：GET /v1/ai/{model}/{task\_id}[[7]](https://docs.freepik.com/api-reference/mystic/get-mystic-task#:~:text=curl%20,key)。根据返回值检查任务状态是否完成。示例返回中，data.generated 字段为结果文件列表[[8]](https://docs.freepik.com/api-reference/mystic/get-mystic-task#:~:text=%7B%20,%5B%20false)，status 由 IN\_PROGRESS 变为 COMPLETED。一旦完成，B 将任务标记为 COMPLETED。
* **结果下载与存储**：任务完成后，B 从 Freepik 返回的 data.generated URL 列表中获取输出（图片/视频）文件。对每个文件，B 从 Freepik 下载后上传至 Cloudflare R2 存储（S3 兼容方式），并将生成的 R2 URL 保存到 tasks.result\_r2\_url 等字段。上传时可根据任务类型和日期分类存储路径，确保文件唯一且可访问。
* **结果回调给 C 站点**：在任务完成并上传 R2 成功后，B 向原 C 站点提供的回调地址发送通知（POST），消息体包括任务状态、Freepik task\_id、以及 R2 存储的文件 URL 等信息。若发生失败（Freepik 返回错误或超时），B 可记录错误详情并通过相同回调通知 C 任务失败状态。回调给 C 站点时可实现重试机制，确保通知到达。
* **错误重试机制**：对于调用 Freepik API 的失败（如网络错误、响应非 200 等），B 应尝试重试（可设定重试次数，例如 3 次），并可切换至备用 API Key 继续请求。如果最终失败，将任务状态标记为 FAILED，并记录错误信息，通知 C 站点任务失败。

## 数据库结构设计（Supabase）

使用 Supabase（PostgreSQL）存储系统核心数据，包括 API Key、任务和调用统计等。主要表结构设计示例：

* **api\_keys**：存储 Freepik API Key 及其状态。字段示例：
* id（主键）、key\_value（API Key 字符串）、name（备注）、daily\_quota（日调用上限）、used\_today（当日已用调用数）、last\_used\_at（最近一次使用时间）、active（启用标志）等。用于轮询选择和监控 Key 使用量。
* **clients**（或 c\_sites）：记录各个 C 站点信息。字段示例：id、name、webhook\_url（C 提供的回调地址）、daily\_limit（可自定义每日调用限额）、used\_today（当日调用量）、rate\_limit（每分钟限流阈值）等。用于给不同客户做隔离限流及统计。
* **tasks**：记录所有转发任务。字段示例：
* id（主键）、client\_id（关联 clients 表）、type（IMAGE\_GEN、VIDEO\_GEN、IMAGE\_EDIT 等）、parameters（JSON，存原始请求参数）、freepik\_task\_id（Freepik 返回的任务 ID）、status（PENDING、IN\_PROGRESS、COMPLETED、FAILED）、api\_key\_id（所用 API Key 外键）、result\_r2\_url（完成后存的 R2 文件 URL）、error\_msg（失败原因）、created\_at、updated\_at 等。
* **calls\_log**（或合并入 tasks）：可记录详细的 API 调用日志，包括请求参数、Freepik 原始响应、调用时间等，便于审计和调试。
* **statistics**（可选）：汇总表或物化视图，按日/月统计调用次数、成功率等。也可通过 Supabase SQL 视图实时查询。

上述设计确保所有任务和调用信息可追踪。Supabase 可利用 SQL 触发器或服务器端函数自动更新统计数据，并支持通过 Admin UI 查询和导出。

## R2 存储处理逻辑

* **结果上传**：任务完成后，B 服务从 Freepik 提供的下载链接中获取文件（图片或视频）。使用 Cloudflare R2 的 S3 API 将文件上传到指定桶中。可以按日期或任务 ID 组织存储目录，如 r2://bucket/tasks/YYYYMMDD/{task\_id}/...。
* **文件命名与路径**：建议为每个文件生成唯一名称（可用任务 ID 和原始文件名），避免覆盖。可以将不同任务类型（图像/视频）分目录存储。
* **URL 访问**：上传完成后形成 R2 公开或签名访问 URL（例如 https://<R2桶域名>/<path>），将此 URL 写入任务记录中供前端和通知使用。建议设置合理的缓存头或过期策略。
* **生命周期管理**：根据业务需求，可对 R2 对象设置生命周期策略（如保留周期、版本管理等），以平衡存储成本和可用性。

## Webhook 接收与转发逻辑设计

* **Freepik 回调接收**：B 服务提供一个专用的 Webhook 接口（如 /api/webhook/freepik），用于接收 Freepik API 的异步回调。该端点解析请求体，根据 data.task\_id 查找对应任务；同时验证请求来源（可检查 HTTP 头或签名，视 Freepik 支持情况）。
* **更新任务状态**：接收到回调后，更新 tasks 表中任务的 status（通常为 COMPLETED 或 FAILED）、记录可能返回的结果 URL 列表等。示例回调 JSON 结构中，data.generated 字段包含输出文件 URL[[8]](https://docs.freepik.com/api-reference/mystic/get-mystic-task#:~:text=%7B%20,%5B%20false)。
* **结果处理**：对于完成状态，B 按前述 R2 逻辑下载并上传文件，然后更新 tasks.result\_r2\_url。对于失败状态，记录错误信息（例如 data.error）并标记任务为 FAILED。
* **通知 C 站点**：在处理完结果后，B 查找任务对应的 clients.webhook\_url，并向其发送 HTTP POST 通知，内容包含任务 ID、最终状态、结果链接等信息。示例：
* {  
   "task\_id": "...",  
   "status": "COMPLETED",  
   "result\_url": "https://r2.example.com/...",  
   "message": "任务已完成"  
  }
* 通知失败时，B 应重试或记录异常。
* **轮询回退**：若 Freepik 的回调不稳定或长时间未收到，可由系统后台定时执行轮询（调用 Freepik GET 状态接口[[7]](https://docs.freepik.com/api-reference/mystic/get-mystic-task#:~:text=curl%20,key)）来主动获取结果并触发上述后续流程。
* **可靠性保障**：对 B 转发给 C 的 Webhook 也应设计幂等或重试机制，确保不遗漏任何通知。必要时可在数据库中记录通知记录，支持再次发送。

## 调用频次统计与限流策略

* **Key 调用统计**：每次调用 Freepik API 后更新 api\_keys.used\_today。每日凌晨或使用时间戳检查机制将所有 Key 的 used\_today 重置为 0。结合 Freepik 文档给出的每日限额[[1]](https://docs.freepik.com/ratelimits#:~:text=,Rate%20per%20seconds)，确保不超过配额。可在数据库或应用层进行限额检查，如当前使用量接近上限则切换 Key。
* **C 站点调用统计**：为每个 C 站点记录日调用数和成功率。可在 clients.used\_today 或 statistics 表中累加。后台管理界面展示各站点调用概况。
* **限流策略**：在 B 站点对外接口处实现速率限制机制（Rate Limiting）。比如对单个 C 站点或 IP 每分钟/每秒限制最大请求数。可使用中间件或数据库计数方式：每当请求到来时，在 clients 表或专门的速率表中增加计数，并对比设定的阈值；超出时返回 HTTP 429 错误。这样防止恶意或失控请求。
* **异常监控**：统计调用失败率或延迟等指标，一旦超过预设阈值（比如失败率高或调用频次突增），触发告警供运维介入。可结合 Supabase 自带的日志或外部监控工具。

## Next.js 服务结构建议

* **项目结构**：利用 Next.js 的 API Routes 和页面功能划分业务模块，示例结构：
* /pages  
   /api  
   /tasks.js # 接收 C 站点任务请求  
   /webhook  
   /freepik.js # 接收 Freepik 回调  
   /admin  
   /tasks.js # 任务管理后台页面  
   /keys.js # API Key 管理页面
* **API 路由实现**：在 /api/tasks.js 中解析 C 站点请求，执行任务入库和转发调用逻辑；在 /api/webhook/freepik.js 处理中转平台回调。业务逻辑可封装为服务模块，调用 Supabase SDK 进行数据库操作，并使用 fetch 或 axios 调用 Freepik API。
* **管理界面**：/pages/admin 下的页面可使用 React 和 Supabase 客户端查询展示数据。譬如，/admin/tasks 显示任务列表和状态，支持按日期或状态筛选；/admin/keys 展示 API Key 使用情况和剩余额度。页面通过 Next.js Server-Side Rendering（getServerSideProps）或静态渲染结合客户端请求实现，保证实时性。此管理界面仅供内部使用，可配合简单认证或 Vercel 的 Password 保护。
* **部署与配置**：可部署在 Vercel、Cloudflare Workers 等支持 Node.js 的平台。将 Supabase 连接信息、Freepik API Key 等保存在环境变量中。Cloudflare R2 的访问凭证也通过环境变量配置 AWS SDK。
* **安全与最佳实践**：后端接口应验证并记录传入请求来源（如 C 站点的身份或密钥），防止未授权调用。对敏感操作（如新增 Key）可限制为内部管理页面使用。使用 HTTPS 通信确保安全。

以上设计方案结合了 Freepik 官方 API 参数[[2]](https://docs.freepik.com/api-reference/mystic/post-mystic#:~:text=,aSDinaTvuI8gbWludGxpZnk)[[4]](https://docs.freepik.com/api-reference/image-to-video/minimax-hailuo-02-768p/post-minimax-hailuo-02-768p#:~:text=,7890.jpg%22%2C%20%22duration%22%3A%206)[[6]](https://docs.freepik.com/api-reference/remove-background/post-beta-remove-background#:~:text=)和调用示例，覆盖了各字段和主要流程，力求满足高可用和可维护的生产级需求。各模块逻辑清晰，可作为开发实现的依据。

**参考资料：** Freepik 官方 API 文档[[2]](https://docs.freepik.com/api-reference/mystic/post-mystic#:~:text=,aSDinaTvuI8gbWludGxpZnk)[[8]](https://docs.freepik.com/api-reference/mystic/get-mystic-task#:~:text=%7B%20,%5B%20false)[[4]](https://docs.freepik.com/api-reference/image-to-video/minimax-hailuo-02-768p/post-minimax-hailuo-02-768p#:~:text=,7890.jpg%22%2C%20%22duration%22%3A%206)[[6]](https://docs.freepik.com/api-reference/remove-background/post-beta-remove-background#:~:text=)[[1]](https://docs.freepik.com/ratelimits#:~:text=,Rate%20per%20seconds)。

[[1]](https://docs.freepik.com/ratelimits#:~:text=,Rate%20per%20seconds) Rate limiting - Freepik API

<https://docs.freepik.com/ratelimits>

[[2]](https://docs.freepik.com/api-reference/mystic/post-mystic#:~:text=,aSDinaTvuI8gbWludGxpZnk) [[3]](https://docs.freepik.com/api-reference/mystic/post-mystic#:~:text=,6489e6daee91%22%2C%20%22status%22%3A%20%22IN_PROGRESS%22) Create image from text - Mystic - Freepik API

<https://docs.freepik.com/api-reference/mystic/post-mystic>

[[4]](https://docs.freepik.com/api-reference/image-to-video/minimax-hailuo-02-768p/post-minimax-hailuo-02-768p#:~:text=,7890.jpg%22%2C%20%22duration%22%3A%206) [[5]](https://docs.freepik.com/api-reference/image-to-video/minimax-hailuo-02-768p/post-minimax-hailuo-02-768p#:~:text=%7B%20%22data%22%3A%20%7B%20%22task_id%22%3A%20%22046b6c7f,) Create a video from text/image - MiniMax Hailuo-02 768p - Freepik API

<https://docs.freepik.com/api-reference/image-to-video/minimax-hailuo-02-768p/post-minimax-hailuo-02-768p>

[[6]](https://docs.freepik.com/api-reference/remove-background/post-beta-remove-background#:~:text=) Remove the background of an image - Freepik API

<https://docs.freepik.com/api-reference/remove-background/post-beta-remove-background>

[[7]](https://docs.freepik.com/api-reference/mystic/get-mystic-task#:~:text=curl%20,key) [[8]](https://docs.freepik.com/api-reference/mystic/get-mystic-task#:~:text=%7B%20,%5B%20false) Get the status of the Mystic task - Freepik API

<https://docs.freepik.com/api-reference/mystic/get-mystic-task>