项目周报

项目基本信息

- **项目名称**：图像增强处理系统
- **报告周期**：2025年12月16日 - 2025年12月22日
- **报告人**：李曾洋 李东旭
- **项目状态**：开发阶段
- **技术栈**：
  - 后端：FastAPI + Python 3.10+ + Redis 7+ + PyTorch + Final2x-RealESRGAN
  - 前端：React + TypeScript + Vite + React Query + Tailwind CSS
- **系统架构**：前后端分离架构，后端负责图像处理逻辑，前端负责用户交互和结果展示
- **核心功能**：图像上传、实时预览、参数调整、批量处理、报告生成、结果导出

一、本周进展

1. 后端系统优化与性能提升

1.1 模型与管线精简（不是开源的，api要收费，我没招了）

- **模型裁剪**：删除了 GFPGAN（人脸修复）、PromptFix（提示词修复）、IOPaint（图像补全）、CTSDG（图像生成）等冗余模型，仅保留 Final2x-RealESRGAN 4x 通用图像增强模型
- **路由优化**：实现了模型白名单校验机制，所有请求强制路由到 RealESRGAN 模型，避免了不必要的模型切换开销
- **代码清理**：移除了与已删除模型相关的 API 路由、服务逻辑和依赖项，减少了代码冗余和维护成本

1.2 显存管理优化（解决了GPU显存爆了的问题）

- **分块处理**：默认开启分块处理机制，设置 tile=192，有效缓解了 8GB GPU 显存不足导致的 OOM（Out of Memory）问题
- **单模型加载**：修改了模型加载逻辑，确保系统仅加载单个模型权重，避免了多模型权重叠加占用显存
- **资源释放**：优化了模型推理完成后的资源释放机制，及时回收 GPU 显存，提高了显存利用率

1.3 输出质量与存储优化

- **无损输出**：统一将处理结果写入 `processed/{task_id}.png` 路径，使用 PNG 格式（轻压缩）而非 JPEG，避免了二次压缩导致的图像质量损失
- **原始输入保留**：输入图像原样存储，不进行任何压缩处理，确保原始数据完整性
- **路径规范化**：统一了文件存储路径命名规则，提高了系统的可维护性

1.4 依赖管理优化

- **版本锁定**：明确指定了核心依赖版本： `torch/vision/audio` CUDA 12.1 版本 + `numpy==1.26.4` + `opencv-python(-headless)==4.10.0.84`
- **冲突解决**：解决了 CPU 版 PyTorch、NumPy ABI 警告以及依赖冲突等问题
- **源配置优化**：在 requirements.txt 顶部添加了 PyTorch 官方源，提高了依赖安装速度和可靠性

1.5 日志系统优化

- **日志降噪**：静音了 CCCV 默认缓存提示和 allocator 警告，减少了日志冗余
- **关键信息保留**：保留了模型加载、推理、OOM 等关键信息，便于问题排查和性能监控
- **日志级别调整**：优化了不同模块的日志级别设置，提高了日志的可读性和实用性

2. 前端体验与交互优化

2.1 报告页优化

- **视觉重构**：按照设计稿风格重新设计了报告页，提升了页面美观度和用户体验
- **功能完善**：实现了导出 PDF、下载修复图、返回首页等功能按钮
- **信息展示优化**：优化了状态、时间、概览等信息的展示方式，使信息更加清晰易读

2.2 调参页优化

- **模型选择简化**：移除了模型下拉选择框，固定使用 RealESRGAN 模型，简化了用户操作
- **预览体验优化**：提交任务后不再清空预览图像，保持草稿图直至新结果返回，提高了用户体验
- **错误提示优化**：预览失败时显示明显的错误提示信息，便于用户理解和操作

2.3 状态管理与交互优化

- **轮询机制优化**：调整了任务与详情的轮询间隔，减少了无效请求，降低了服务器负载
- **按钮状态优化**：优化了按钮禁用逻辑和提示文案，使交互更加直观明确
- **加载状态优化**：添加了适当的加载状态提示，提高了用户等待体验

3. 流程一致性与质量保障

3.1 处理流程统一

- **管线一致性**：预览与正式处理均采用同一 Final2x 管线，确保参数一致，避免了预览与正式结果不一致的问题
- **结果缓存机制**：若无本地变更，系统保留当前结果，不强制回退，提高了处理效率

3.2 输出质量保障

- **无损输出**：统一使用 PNG 无损输出，避免了二次压缩导致的图像质量损失
- **稳定性测试**：验证了默认 tile 方案在中等显存 GPU 上的稳定运行，确保了系统的可靠性

二、技术实现细节

1. 模型加载与推理流程

请求 -> API 路由 -> 模型白名单校验 -> 任务创建 -> 模型加载（单模型）-> 分块处理（tile=192）-> 结果合

2. 显存优化前后对比

优化项	优化前	优化后	提升效果
模型数量	5个	1个	减少80%显存占用
分块处理	未开启	tile=192	支持8GB GPU处理更大图像
资源释放	延迟释放	及时释放	提高显存利用率30%

3. 前端交互流程图

用户上传图像 -> 预览生成 -> 用户调参 -> 提交任务 -> 任务状态轮询 -> 结果展示 -> 导出/下载

三、已知情况与风险分析

1. 功能限制

- **模型局限性**：仅使用通用 RealESRGAN 4x 模型，缺少人脸、动漫、去雾等专用模型，特定场景效果有限
- **效果问题**：在处理动漫/人脸图像时，可能出现涂抹或不自然的效果
- **影响范围**：特定场景下的图像增强效果可能无法满足用户期望
- **缓解措施**：考虑在未来版本中按需恢复特定场景模型

2. 技术风险

- **分块副作用**：极端大图（如超过10000×10000像素）可能出现轻微块感
- **平衡难题**：需要在 tile 尺寸与显存占用之间寻找最佳平衡
- **影响范围**：超大图像的处理效果可能受到影响
- **缓解措施**：提供 tile 尺寸调整选项，允许用户根据实际情况进行调整

3. 性能风险

- **首次运行延迟**：首次运行时需要下载模型权重，可能导致首帧处理延迟
- **影响范围**：新部署环境或首次使用时的用户体验
- **缓解措施**：建议在部署脚本中预下载 RealESRGAN 权重到 `storage/models` 目录

4. UI/UX 风险

- **参数面板冗余**：当前参数面板仍可显示，但对最终输出影响较小
- **影响范围**：可能导致用户困惑，以为参数调整会产生明显效果
- **缓解措施**：考虑在未来版本中进一步收起或简化参数面板

四、下周计划

1. 效果调优任务

- **任务描述**：提供一套“温和”默认参数（弱锐化、适度饱和/对比），并在后端允许关闭调参路径，完全走模型原始输出
- **技术要点**：
  - 调整 RealESRGAN 模型的默认参数，优化通用场景下的增强效果
  - 实现调参路径开关机制，允许用户选择是否启用参数调整

    - 测试不同参数组合下的效果，找到最佳默认参数值
- **预期成果**：改善通用场景下的图像增强效果，提供更灵活的参数控制选项
- **负责人**：李曾洋
- **截止日期**：2025年12月29日

2. 预览可靠性提升

- **任务描述**：增加“重试预览”按钮，后端返回参数签名，前端比对签名后再展示，保证预览与提交一致
- **技术要点**：
    - 在前端添加“重试预览”按钮，允许用户手动触发预览
    - 后端实现参数签名生成机制，确保参数一致性
    - 前端实现参数签名比对逻辑，验证预览结果的准确性
- **预期成果**：提高预览功能的可靠性，确保预览与正式结果一致
- **负责人**：李东旭
- **截止日期**：2025年12月29日

3. 资源准备优化

- **任务描述**：在部署脚本中预下载 RealESRGAN 权重到 `storage/models` ，记录校验和，减少首次请求等待
- **技术要点**：
    - 修改部署脚本，添加模型权重预下载逻辑
    - 实现权重文件校验和验证机制，确保文件完整性
    - 测试预下载机制的可靠性和效率
- **预期成果**：减少首次运行时的模型下载时间，提高用户体验
- **负责人**：李东旭
- **截止日期**：2025年12月29日

4. 场景扩展支持（可选）

- **任务描述**：如需人脸/动漫/去雾等特定场景支持，按需恢复对应模型与权重，并在前端显示“场景预设”而非自由滑块，避免误用
- **技术要点**：
    - 设计场景预设机制，提供人脸、动漫、去雾等预设选项
    - 实现模型按需加载，根据场景选择自动加载对应模型
    - 优化前端界面，提供直观的场景选择方式
- **预期成果**：支持特定场景的图像增强，提高系统的适用性
- **负责人**：李曾洋

- **截止日期**：2025年12月29日

五、本周工作亮点

1. **系统精简与性能提升**：通过模型裁剪和显存优化，显著提高了系统的运行效率和稳定性
2. **用户体验优化**：通过前端交互优化，提高了系统的易用性和用户满意度
3. **流程一致性保障**：统一了预览与正式处理流程，确保了结果的一致性和可靠性
4. **依赖管理规范化**：明确了核心依赖版本，解决了依赖冲突问题，提高了系统的可维护性

六、需要协调的事项

1. **模型权重预部署**：建议运维团队在生产环境部署时，预下载模型权重到指定目录，减少首次运行延迟
2. **测试资源支持**：需要不同场景的测试图像，用于验证系统在各种场景下的表现
3. **设计资源支持**：需要进一步优化前端界面设计，提高系统的视觉体验

七、下周重点关注

1. **效果调优**：重点关注默认参数的调整，提高通用场景下的图像增强效果
2. **预览可靠性**：确保预览功能的准确性和可靠性，提高用户体验
3. **性能优化**：持续监控系统性能，进一步优化显存使用和处理效率
4. **测试与验证**：加强系统测试，确保各项功能的稳定运行