yiwuyi 产品需求文档.

（一期：能用就好）xD

字不好看dbq.

这是一个什么东西呢？

能够 离线下载 在线视频网站的视频 的 WebApp

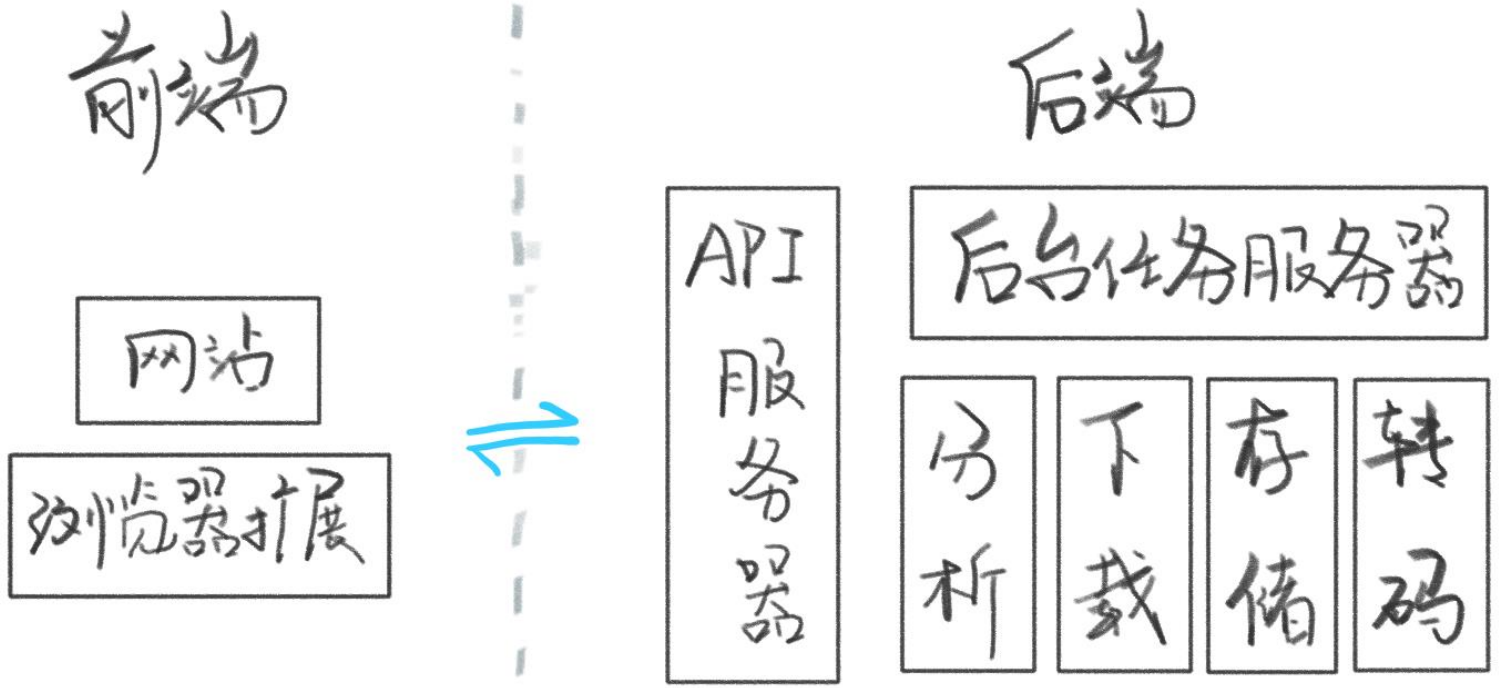
（就像百度云离线下载
（只不过是针对在线视频网站的
（是的我就钦定 WebApp 了

它有什么功能呢？

- 能够接收用户发来的视频页面地址
  - 能够分析这个视频地址
  - 能够下载对应的视频并存储在云端
    - 能够让用户下载到服务器下完了的视频
    - 能够让用户把服务器下完了的视频转存到用户的网盘（待定）
    - 能够让用户直接在线播放服务器下完了的视频
      - 用户还可以选择清晰度（也就是说 能够对下完了的视频进行转码）

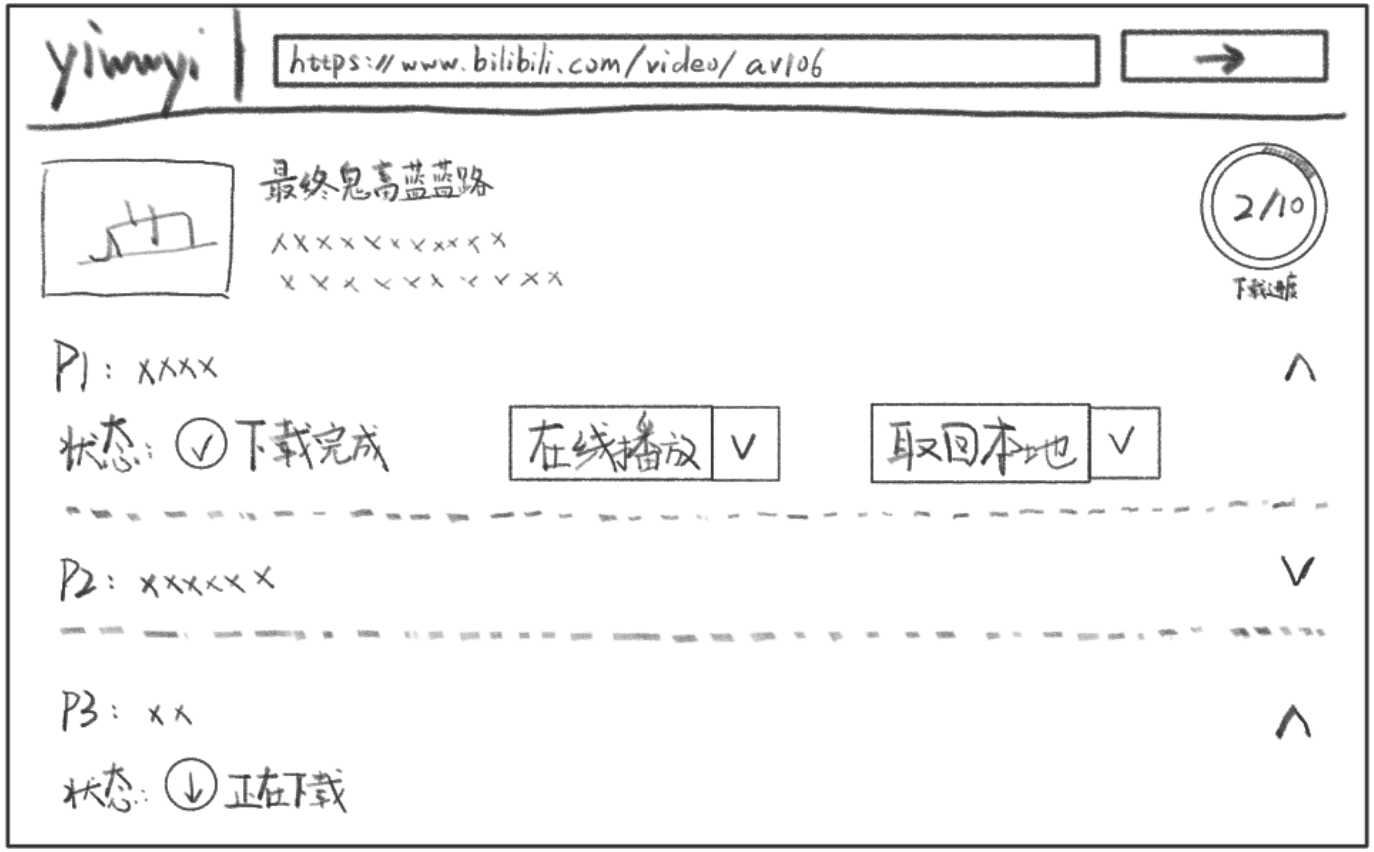
- 能够接收用户发来的直播间页面地址
  - 能够分析这个直播间地址
  - 能够监控对应的直播间
    - 一旦开播 能够开始录制（下载）直播视频信号并存储在云端
      - ……（同上）

基本架构



前端 网站 示例

灵魂绘画仅用于说明功能需求，不涉及设计

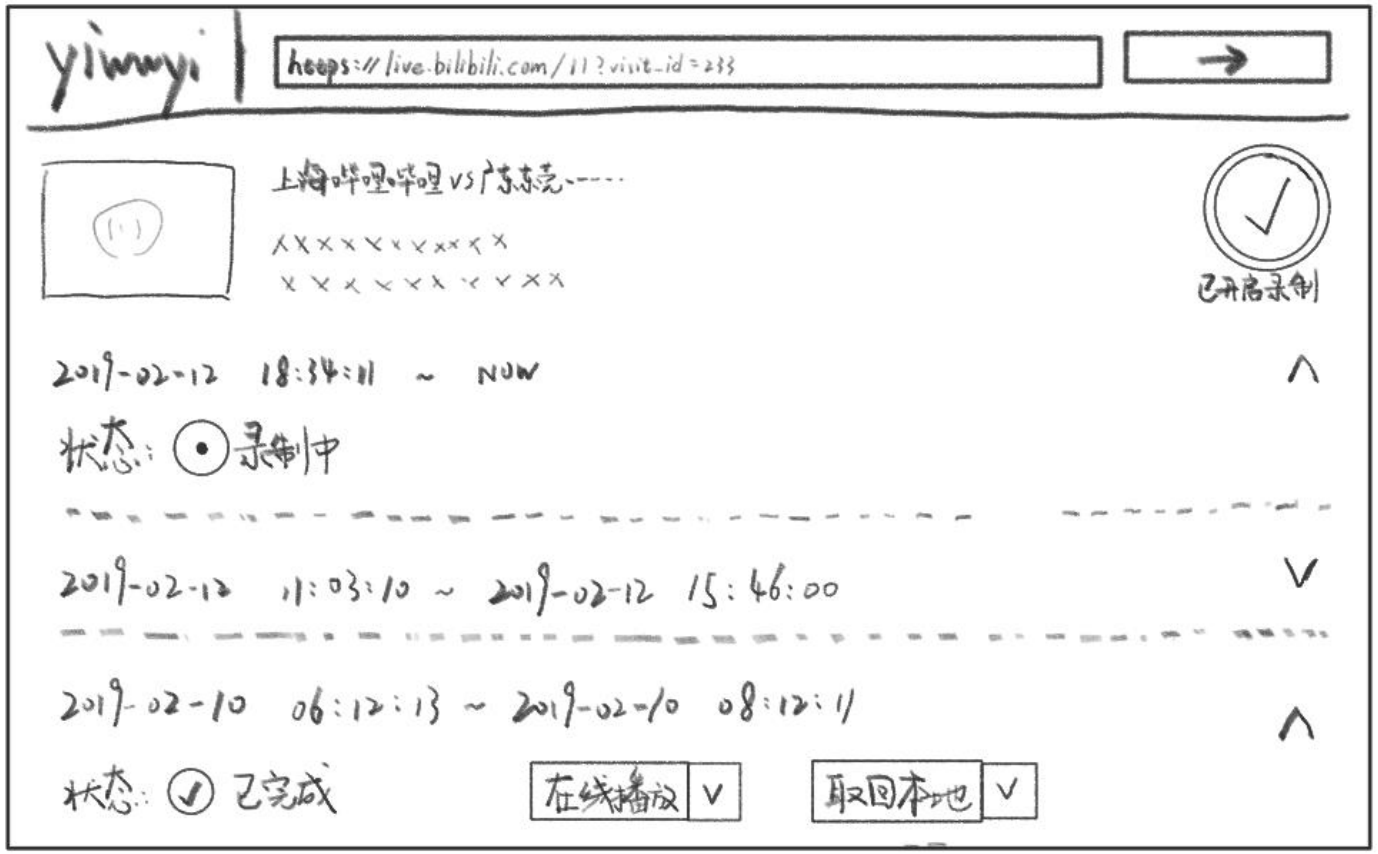


接收用户输入的网址，
并从后端获取对应视频信息

如果一个网址对应多个视频
(如分P)，则呈现每一视频
对应的状态、在线播放按钮、
取回本地按钮。

前端 网站 示例

灵魂绘画仅用于说明功能需求，不涉及设计



接收用户输入的网址，
并从后端获取对应直播信息

类似普通视频
每次直播视为一个分P

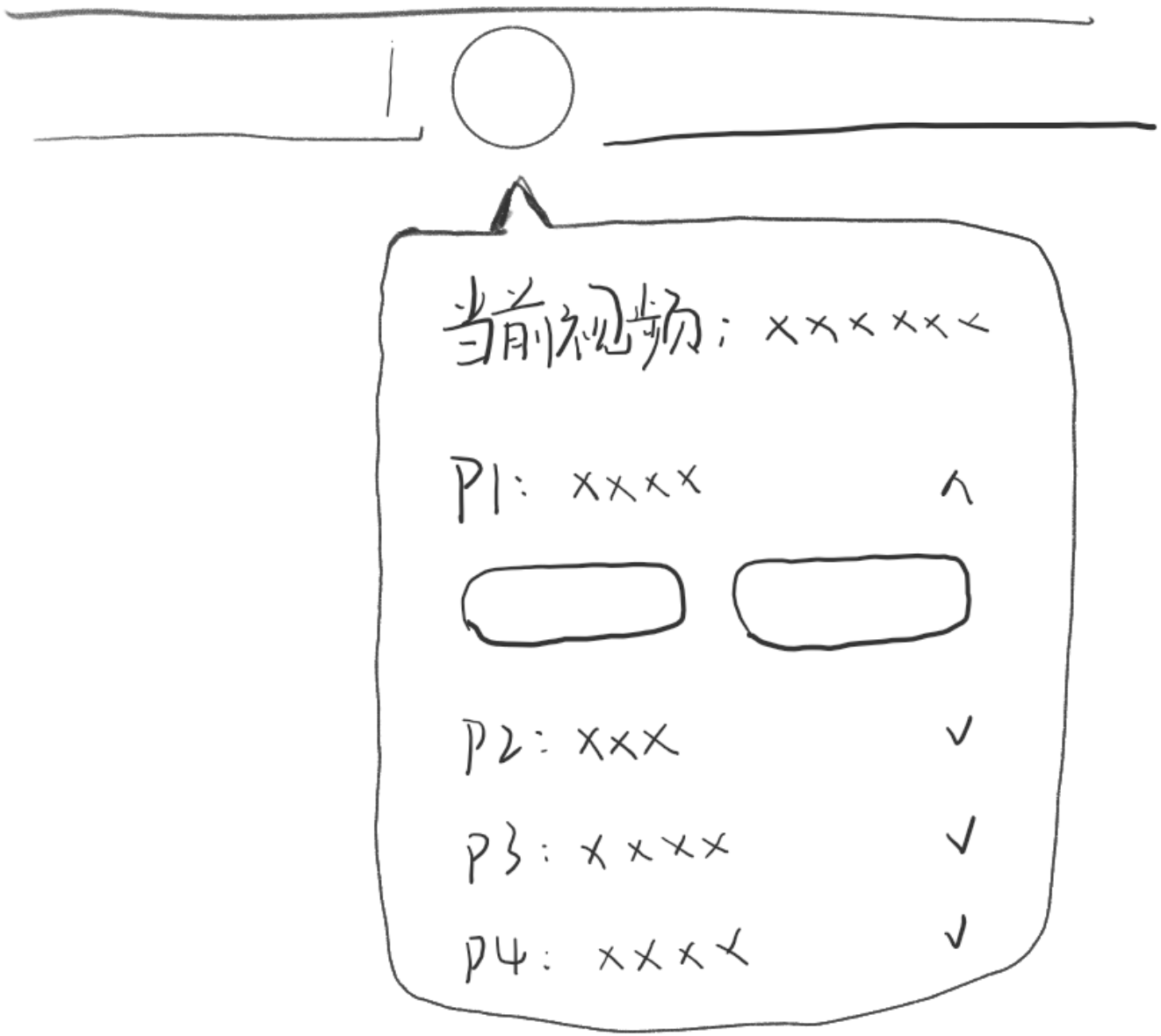
前端
浏览器插件
示例

灵魂绘画仅用于说明功能需求，不涉及设计

* 可获取当前页面URL
  并向后端请求对应视频数据

* 简略显示当前视频信息
  给出下载状态与在线播放、取回本地按钮

* 可展示跳往yiwwyi网站对应页面的按钮



一个视频的生命历程

获得视频信息

- 检查数据库中是否有记录
  - 若有，则直接返回
  - 若无，则使用对应视频网站的API查询信息，并存入数据库

添加视频下载任务

- 若视频已下载至服务器
  - 无需再次添加
- 若视频尚未下载至服务器，则向后台任务服务器下发下载任务
  - 后台任务服务器定时向API服务器汇报任务进度

在线播放/本地取回

- 若用户选择服务器上已有的画质
  - 直接返回
- 若用户请求其他画质
  - 向后台任务服务器下发转码任务
    - 后台任务服务器定时向API服务器汇报任务进度

视频文件的自动删除

- 若超过一定时间，没有用户请求播放/下载某视频文件
  - 则将该视频文件占用的空间释放

后端 API 服务器接口 基本要求

1. getVideoInfo (video_url/video_id)

返回对应网址或 video_id 的视频在 yiwwyi 服务上的信息

若视频网址合法，返回值应至少包括：

```
video_id: 123456,                  // 该视频在 yiwwyi 上的唯一 ID
is_live: false,                    // 是否为直播视频
videos: [ //
    {
      video_file_id: 12345677 // 该视频分P在 yiwwyi 上的唯一 ID
      download_status: ``      // 尚未下载/正在下载/下载完成,
      transform_status: [],    // 各种画质尚未转码/正在转码/转码完成
      url: [],                 // 已有的各种画质对应的网址
    }, ...
]
```

后端 API 服务器接口 基本要求

2. addVideoDownload (video_id/video_file_id)

添加 video_id 对应所有视频或 video_file_id 对应视频的下载任务

返回值应至少包括:

```
is_success: false, // 添加是否成功
err_msg: xxx // 失败时的错误信息
```

后端 API 服务器接口 基本要求

3. addVideoTransform (video_id/video_file_id, transform_type)

请求转码 video_id 对应所有视频或 video_file_id 对应视频至指定类型transform_type

若video_id合法，返回值应至少包括:

```
is_success: false, // 添加是否成功
err_msg: xxx // 失败时的错误信息
```

注：以上所涉及的所有接口名称、参数名称、返回值名称、返回值格式仅用于辅助说明“后端至少需要实现哪些类型的接口”以及“这些接口至少需要做到什么”，具体接口格式由前后端根据项目需求及进度自行协商制定。

后端各个模块之间的关系（有一些东西前面的图没画出来）

1. **API 服务器**是一个基于 Koa2 的 Web 服务器，它负责
    1. 听用户使唤
    2. 跟数据库打交道
    3. 向合适的后台任务服务器发送请求并获取结果
    4. 向用户通报后台任务服务器的结果

后端各个模块之间的关系（有一些东西前面的图没画出来）

2. **后台任务服务器**是一个统称，

   它既可以是提供分析服务的，也可以是提供下载服务的，也可以是提供…服务的。

   1. 分析服务：接收发送过来的视频 URL，跟视频网站打交道，得到对应视频的信息并返回给 API 服务器
      1. You may consider a you-get wrapper
   2. 下载服务：接收发送过来的视频文件 URL 与存储位置，下载对应的视频文件，并定期向 API 服务器汇报进度
      1. 需要支持多个文件同时下载，需要支持断点续传
      2. 听起来很简单，但独立出来是便于添加其他功能，比如切片
   3. 转码服务：接收发送过来的原视频文件位置及转码格式，调用 ffmpeg 对其进行转码，并定期向 API 服务器汇报进度

后端各个模块之间的关系（有一些东西前面的图没画出来）

2. **后台任务服务器**是一个统称，

   它既可以是提供分析服务的，也可以是提供下载服务的，也可以是提供…服务的。

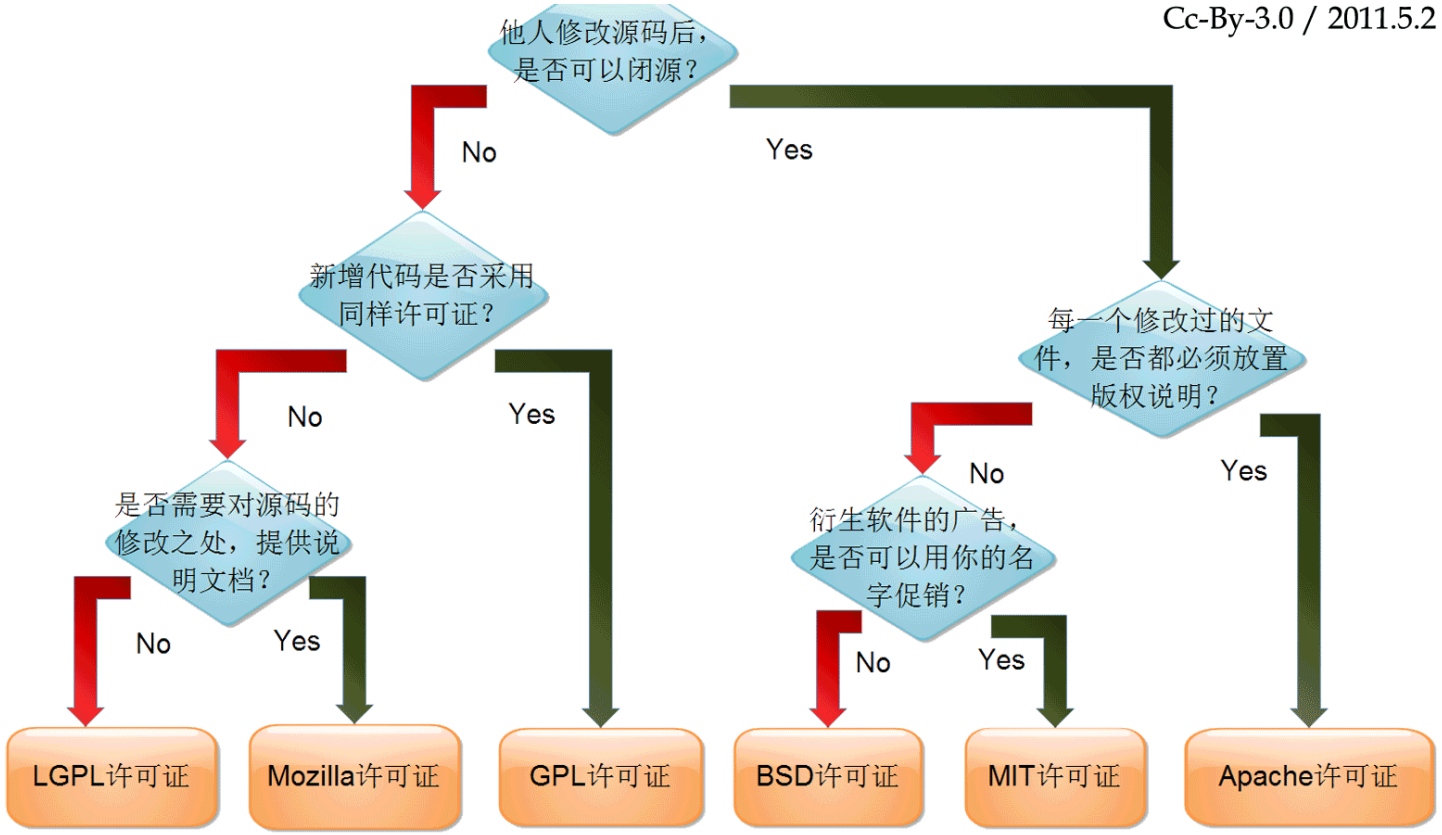
   1. 存储服务：定期巡查过期视频文件并删除
      1. 听起来很简单，之所以独立出来是因为二期的存储服务事务比较多
   2. 监听服务：接收指定的直播房间，与对应的弹幕服务器建立连接，收到开播信息时通知 API 服务器触发下载任务

后端的补充说明

1. 由于部分任务比较简单，所以也可以考虑
    1. 将“分析”服务、定期清除过期视频文件 合并至 API 服务器内
    2. 将下载服务、转码服务等合并为同一个服务器
2. 各个模块的接口与通讯方式由后端成员自行决定
3. 前面将各个模块分得这么细，主要是便于后期在生产环境上多个服务器进行个别部署
    1. 比如有的服务器带宽大但硬盘小，就适合只部署下载服务而不利用其进行存储

技术栈 & 编码规范约定

1. 本项目前后端均使用 **JavaScript** 语言。
2. 前端、后端采用何技术栈由对应小组自行决定。
3. 数据库由后端自行选择使用 MySQL/PostgreSQL/MongoDB。
   （为容器化后共享数据考虑，暂不使用SQLite）
4. 前端、后端的数据交互使用 RESTful API，必要时可用 WebSocket。
5. （推荐）使用 ESLint 规范代码，使用 Airbnb 配置。
   （便于他人查看代码帮你修bug）



开源协议

1. 本项目使用 MIT 许可证。
2. 在引用其他项目的代码时，注意其开源协议的要求，谨慎使用采用 GPL 许可证的项目的代码。

暂定分工

结合前期在 GitHub 上的调查，暂定分工如下（GitHub 用户名）

1. 前端：und3fin3 robinWongM Simonacb
2. 后端：laoyeche1998 charlzhg lmtiga hygodlike
3. 开发服务器运维：charlzhg

非最终名单，可调换
请各组根据需要建立以 yiwwyi- 为开头的 Repository 并放置代码
各组可建立微信群等以便组内沟通

yiwuyi 产品需求文档.

(二期：要跑生产环境的！) xD

字不好看dbq.

二期目标（画大饼）

1. 建立用户系统
    1. 使用第三方网站账号即可登录
    2. 用户可收到订阅的直播间提醒
2. 后端服务器容器化
    1. 不因为其中任何一个服务器宕机而导致服务不可用
    2. 使 下载服务 能有冗余备份，避免直播间录制时出现缺斤少两的情况
    3. 针对化部署，使每个物理服务器可以物尽其用
    4. 提升请求与任务的吞吐量
3. 更加长效的存储空间管理机制
4. ……