**项目开发文档：类小红书短视频网页**

**1. 项目概述**

本项目是一个模仿小红书风格的图文、短视频分享网页应用。用户可以在主页通过瀑布流的形式浏览不同分类下的图文和视频内容，并可以进行搜索和分类筛选。项目还提供了一个独立的发布页面，用户可以通过该页面上传视频或图文，填写相关信息，并生成对应的数据条目。

此项目旨在实现一个内容分享平台的前端展示和交互逻辑，为用户提供一个美观、易用的内容消费和创作界面。

**2. 技术栈**

本项目主要采用了现代前端技术栈，具体如下：

**HTML5:** 作为网页的基本结构语言。

**CSS3:** 用于定义页面的样式，包括布局、颜色、字体等，并使用了 CSS 变量（`var()`）来管理全局样式，实现了灵活的样式配置。

**JavaScript (ES6):** 作为主要的编程语言，负责实现页面的动态交互和逻辑功能。

**Vue.js (v3):** 一个渐进式的 JavaScript 框架，用于构建用户界面。本项目通过 CDN 引入 `vue.global.prod.js`，并使用其 `createApp`、`ref` 和 `computed` 等核心 API 来实现响应式数据绑定和视图更新。

**3. 项目结构**

**本项目由以下几个核心文件组成：**

**`index.html`:** 主页面，即内容发现页。

**布局:** 采用经典的“左侧边栏 + 右侧主内容区”布局。

**功能:**

顶部包含 Logo、搜索框和链接。

内容区上方是分类标签导航。

核心内容通过瀑布流（CSS `column-count`）展示，实现了响应式布局，可根据屏幕宽度自动调整列数。

通过 Vue.js 动态渲染 `videos.json` 中的数据。

**`publish.html`:** 内容发布页面。

**布局:** 同样采用“左侧边栏 + 右侧主内容区”的布局。

**功能:**

提供“上传视频”和“上传图文”的切换标签页。

支持拖拽文件或点击按钮上传。

提供预览区，实时显示上传的文件。

包含一个表单，用于填写标题、分类、作者等信息。

可以一键生成符合格式的 JSON 数据条目，并提供复制到剪贴板的功能。

**`videos.json`:** 项目的数据源文件。

**格式:** 一个 JSON 数组，每个对象代表一个发布条目（视频或图文）。

**字段:**

`id`: 唯一标识符。

`category`: 内容分类（如：推荐、穿搭、美食）。

`type`: 内容类型（`video` 或 `image`）。

`src`: 媒体文件的路径。

`title`: 标题。

`author`: 作者昵称。

`avatar`: 作者头像的 URL。

`likes`: 点赞数。

`tag`: 附加标签（如：星际穿越、深夜放毒）。

**4. 功能说明**

**主页 (`index.html`)**

**1. 动态数据加载:**

页面加载时，会通过 `fetch` API 异步请求 `videos.json` 文件。

若请求成功，则将获取到的数据渲染到页面上；如果请求失败或数据格式不正确，则会调用 `makePosts()` 函数生成一组本地的 mock 数据作为备用，保证页面的正常展示。

**2. 内容展示:**

以卡片形式在瀑布流中展示内容。

根据条目的 `type` 字段，自动渲染为 `<video>` 或 `<img>` 标签。

卡片下方会显示标题、作者信息（头像和昵称）以及点赞数。

**3. 分类筛选:**

用户可以点击顶部的分类标签（如：推荐、穿搭、美食等）来筛选内容。

通过 Vue.js 的计算属性 `filtered` 实现，该属性会根据当前激活的 `activeTab` 过滤出相应分类的内容。

**4. 关键词搜索:**

用户可以在顶部的搜索框内输入关键词。

`filtered` 计算属性同样会根据 `searchQuery` 的值对标题或作者进行模糊匹配，实时更新显示的结果。

**5. 点赞功能:**

每个卡片右下角都有一个点赞按钮。

点击后，对应帖子的 `likes` 数量会加 1，并实时更新视图。这是一个纯前端的交互效果。

**发布页 (`publish.html`)**

**1. 上传方式切换:**

用户可以通过点击“上传视频”和“上传图文”标签来切换上传模式。

切换后，文件选择器 `accept` 属性和上传按钮的文案会相应地改变。

**2. 文件上传与预览:**

支持两种文件上传方式：

点击“上传”按钮，从本地文件系统中选择。

直接将文件拖拽到上传区域。

文件上传后，会在预览区生成对应的 `<video>` 或 `<img>` 标签进行预览。

**3. JSON 条目生成:**

用户在表单中填写完标题、分类、作者等信息后，可以点击“生成 JSON 条目”按钮。

JavaScript 会根据当前的状态（`state`）和表单输入值，自动生成一个符合 `videos.json` 格式的 JSON 对象。

生成的内容会显示在页面下方的文本框中。

**4. 复制到剪贴板:**

生成 JSON 条目后，用户可以点击“复制到剪贴板”按钮。

该功能利用了浏览器的 `navigator.clipboard.writeText` API，可以方便地将生成的 JSON 文本复制，以便手动粘贴到 `videos.json` 文件中。

**5. 使用说明**

**1. 环境准备:**

无需复杂的后端环境，只需一个支持静态文件服务的 HTTP 服务器即可。可以使用 VS Code 的 `Live Server` 插件，或者通过 Node.js 的 `http-server` 等工具。

**2. 本地运行:**

将 `index.html`、`publish.html` 和 `videos.json` 文件以及存放视频和图片的文件夹（如 `video/`, `picture/`）放置在同一个项目根目录下。

通过 HTTP 服务器启动该项目。例如，使用 Live Server 右键点击 `index.html` 选择 "Open with Live Server"。

在浏览器中打开对应的地址（如 `http://127.0.0.1:5500/index.html`）即可看到主页。

**3. 发布新内容流程:**

在浏览器中访问 `publish.html` 页面。

上传你的视频或图片文件。

在右侧表单中填写内容的详细信息。

点击“生成 JSON 条目”按钮。

点击“复制到剪贴板”按钮。

打开项目中的 `videos.json` 文件，将复制的 JSON 对象粘贴到数组的末尾（注意，需要手动在上一条目的 `}` 后添加一个逗号 `,`）。

刷新 `index.html` 页面，即可看到新发布的内容。