其实项目过程是有很多bug的但是没有做实时记录，现在只能写出我个人认为比较费事的问题了

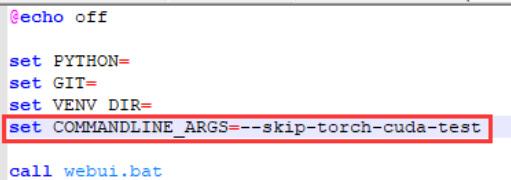
Sd本地部署问题

由于sd Stable-Diffusion对电脑的配置（显卡、内存）要求比较高，是核心的要点，其中显卡、内存、硬盘和cpu是至关重要的元素。在这些要素中，显卡的选择尤为关键。由于我是新手，选择使用整合包一键部署。但是部署的前提使用英伟达的显卡在多少教学中都没有提出，还需要安装驱动，没有办法只能手动部署。

问题1

解压stable-diffusion-webui-master，双击webui-user.bat运行，它会弹出一个CMD终端，自动进行安装和下载相关依赖，这里下载依赖也很麻烦，每次下载tensorflow下载一半都会出现连接异常，下得又慢，2GB每次都预计一小时。我重下了不下5次。过段时间之后将会出现“GPU集显的检查”的报错问题

解决方法：编辑webui-user.bat中COMMANDLINE\_ARGS值为–skip-torch-cuda-test



问题2

当webUI打开时，我立刻点击生成图片，结果缺少模型



其实这个问题很简单，下一个模型就行了。但是因为我是入门，根本没有这方面的概念，我不知道checkpoint文件是什么，我以为只能提供作者的模型。所以我从网盘下载（500kb/s）连模型都画风都不知道就下载了近50GB的模型。

问题3

放入模型生成图片，又报了一个错



因为缺少clip-vit-large-patch14依赖包，需要从huggingface.co网站下载，而大陆用户连接不上huggingface.co，因此会导致运行报错

解决方法：通过魔塔社区镜像站获取。Clone地址为：https://www.modelscope.cn/AI-ModelScope/clip-vit-large-patch14.git



到此就能正常使用了

Python后端

问题1

当我尝试将sd返回的图片通过二进制数据存到数据库mysql中，告诉我图片过大

解决方法

将字段类型改为longBlob

Vue前端

问题1

**Access to fetch at '目标 URL' from origin '源 URL' has been blocked by CORS policy**：当浏览器向一个不同源（协议、域名、端口中任意一个不同即为不同源）的服务器发送 fetch 请求（例如使用 fetch API 进行数据获取），并且服务器没有配置 CORS 头信息来允许该源的请求时，就会出现这个错误。这是浏览器出于安全策略，阻止了跨域请求的响应。

解决方法：

在 Python 的 Flask 应用中，可以使用 flask\_cors 扩展。先安装 flask\_cors：pip install flask-cors ，然后配置如下

from flask import Flask

from flask\_cors import CORS

app = Flask(\_\_name\_\_)

CORS(app)

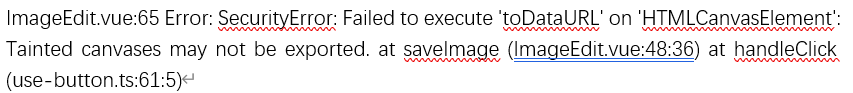
# 其他路由和处理逻辑

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

问题2

当我在画布中完成二次创作时，点击保存，报错



是因为在绘制到画布的图像数据来源是跨域的。例如，你从一个不同域名的服务器获取了图片，并将其绘制到了画布上，浏览器出于安全考虑，禁止从受污染的画布中导出数据。

解决方法：

onMounted 生命周期钩子函数中，img.crossOrigin = 'anonymous'; 这行代码设置了 Image 对象的 crossOrigin 属性为 'anonymous'。这告诉浏览器以匿名方式跨域加载图片，即不会带上用户的身份信息

问题3

当我使用橡皮擦功能时，发现在把笔记擦掉的同时，还把背景生成的画布也擦了

解决方法

使用两层画布，使他们重叠，笔记只对第二层画布生效