**数字内容安全**

**实验报告**

****

**姓 名 詹冲**

**学 号 2023211616**

**指导教师 杨震**

**学 院 网络空间安全学院**

**2025年 5 月 30 日**

实验名称 垃圾邮件过滤系统 实验日期： 5月30日 指导老师 杨震 得分

学院 网络空间安全学院 专业 信息安全 班次 2023211801 姓名 詹冲 学号 2023211616

1. **实验目的**

（1）掌握在Linux下安装和使用版权保护区块链系统

（2）掌握版权保护区块链系统主要功能模块

（3）区块链的原理

1. **实验内容**

（1）分析并调试版权保护区块链系统程序主要功能模块

（2）选取测试实验数据（txt文本、mp3音频、jpg图像）

（3）运行Linux下的版权保护区块链系统

（4）用版权保护区块链系统对实验测试数据提取版权信息进行不重复的上链。

1. **系统整体描述和分功能描述**

**（1）系统整体描述**

该系统是一个基于区块链技术的数字版权保护平台，支持音频、文本和图像三种类型的内容原创性验证与版权登记。系统通过区块链记录每一笔交易（即版权注册），利用音频指纹、文本相似度、图像相似度等技术验证内容的原创性，防止重复或侵权内容进入区块链。

系统由两大主要模块组成：

1.区块链核心模块：负责区块、交易的创建、验证与存储，维护整个区块链数据结构。

2.Web服务模块（基于Flask）：提供前端交互接口，包括上传文件、内容查重、版权登记（发布交易）、区块链浏览等。

**（2）分功能描述**

**1.区块链核心功能**

①交易管理：定义版权交易数据结构，封装版权内容的关键信息。

用到的类和函数：

* classTransaction：封装交易的属性（title、filename、author、public\_key、genre、media）
* Transaction.\_\_init\_\_()：初始化交易对象
* Transaction.toDict()：将交易对象转换为字典形式，方便存储与传输
* Transaction.\_\_str\_\_()：交易对象字符串表示，方便调试和打印

②区块管理：定义区块数据结构，封装包含交易的区块信息。

用到的类和函数：

* classBlock：区块类，包含索引、时间戳、前一区块哈希、交易内容
* Block.\_\_init\_\_()：初始化区块
* Block.compute\_hash()：计算区块的哈希值，作为区块链的链式结构关键
* Block.serialize()：将区块信息序列化成字典，方便存储或网络传输

③区块链操作与维护：管理区块链，负责添加、验证区块，进行原创性校验，持久化区块链状态等。

用到的类和函数：

* classBlockchain：区块链主类，管理整个链和未确认交易
* Blockchain.\_\_init\_\_()：初始化空链和未确认交易
* Blockchain.create\_genesis\_block()：生成创世区块，链的起点
* Blockchain.new\_transaction()：创建新交易，保存到未确认交易中
* Blockchain.mine()：根据未确认交易生成新区块并添加到链上
* Blockchain.verify\_block(block)：验证区块合法性，包含前一区块哈希验证及内容原创性检测（调用音频、文本、图像查重接口）
* Blockchain.lookup(transaction)：查找链中是否存在与指定交易相似或重复的内容
* Blockchain.add\_block(block)：将新区块添加至链并持久化（保存到文件）
* Blockchain.check\_integrity()：链完整性检测（目前未实现，返回0）
* Blockchain.last\_block（属性）：返回链上最新的区块

**2.Web服务功能（Flask应用）**

①系统初始化及目录管理：初始化区块链实例，加载已有链数据，创建所需文件夹。

用到的代码及流程：

* 读取./blockchain/chain.pkl文件恢复区块链状态
* 创建目录./blockchain、./uploads、./tmp（如果不存在）

②前端页面路由：提供基本的网页访问路由，渲染主页、区块链展示页、关于页和FAQ页。

用到的路由函数：

* /→index()：主页
* /blockchain→blockchain\_page()：区块链数据展示页面
* /about→about\_page()：关于我们页面
* /faq→faq\_page()：常见问题页面

③静态文件访问：提供访问上传文件的接口（支持指定文件路径）

用到的路由函数：

* /uploads/<path:filename>→custom\_static(filename)：根据上传文件名提供访问

④文件上传与版权查重/发布接口：处理用户上传的文件，支持查重（lookup）和发布（publish）两种操作。

用到的路由函数：/upload（POST）→ upload()

* 检测请求是否包含文件contentFile，并计算文件的SHA256值作为唯一文件名。
* 根据action参数执行不同操作：
* lookup：将文件保存至临时目录tmp/，调用blockchain.lookup()检查是否存在相似版权内容，返回查重结果后删除临时文件。
* publish：判断文件是否已存在于uploads/目录，防止重复上传。若不存在，则保存文件，创建新交易，调用blockchain.new\_transaction()和blockchain.mine()进行上链。挖矿失败则删除文件并返回失败信息，成功则返回包含区块信息的响应。

⑤区块链数据访问接口：提供区块链数据的API接口，供前端查询。

用到的路由函数：

* /chain(GET)→get\_chain()：返回当前完整链的数据（JSON格式）
* /mine(GET)→mine\_unconfirmed\_transactions()：对当前未确认交易进行挖矿，生成新块并返回

1. **实验步骤、结果及分析**

**（1）分析并调试版权保护区块链系统程序主要功能模块**

该版权保护系统以Flask框架构建Web服务端，配合自定义的区块链结构，支持多媒体数据的版权上链和原创性验证。主要模块如下：

**1.区块链核心模块**

①Transaction类：

* 用于定义一次版权登记操作的数据结构，字段包括标题、原文件名、作者、公钥、媒介类型、媒体文件名。
* 提供toDict()方法用于结构化存储。

②Block类

* 封装一个区块，包含索引、时间戳、前一个区块的哈希值、交易信息。
* 提供compute\_hash()方法进行哈希计算。

③Blockchain类

* 管理区块链数据结构，包含：
* 区块链链条self.chain
* 待确认交易self.unconfirmed\_transactions
* 提供核心方法：
* new\_transaction()：添加待上链交易。
* mine()：验证并生成新区块。
* verify\_block()：查重逻辑，根据媒介类型调用相应算法判定原创性。
* lookup()：用于查找媒体是否已存在于链上。
* add\_block()：将新区块添加到链并序列化保存。

**2.文件查重模块**

①音频比对模块

* 使用acoustid和chromaprint提取音频指纹。
* 通过比特数不同的个数计算相似度。
* 相似度大于0.9视为非原创。

②文本查重模块

* 利用TLSH哈希对文本文件进行比对。
* 差异值小于100视为非原创。

③图像比对模块

* 使用image-match提取图像特征签名，计算归一化距离。
* 相似度小于0.4视为非原创。

**3.FlaskWeb应用服务端**

①提供前端界面index.html,blockchain.html,aboutus.html,faq.html等。

②提供后端接口：

* /upload：文件上传与版权查重、登记处理逻辑。
* /mine：触发区块挖矿。
* /chain：返回区块链数据。
* /uploads/<filename>：提供已上传文件访问。

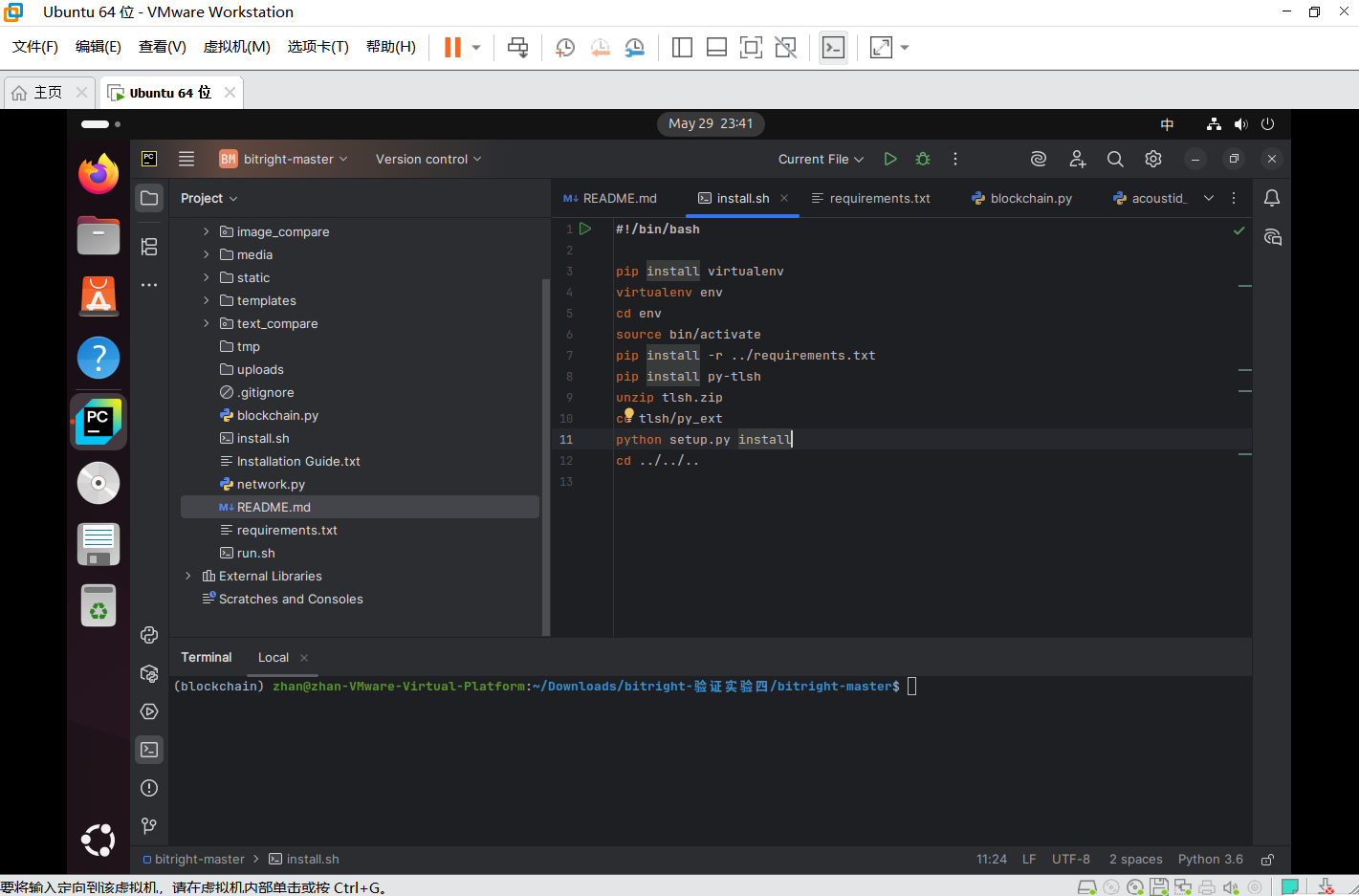
**（2）选取测试实验数据**

本实验选取以下类型样本文件用于测试：

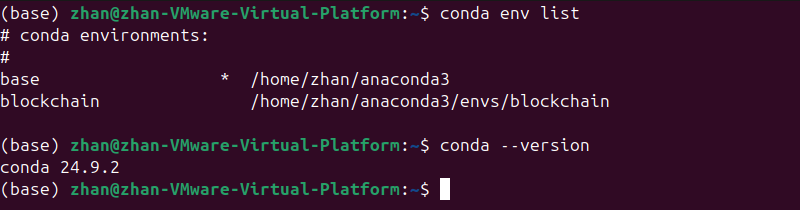
|  |  |  |
| --- | --- | --- |
| 类型 | 文件名 | 描述 |
| 文本 | doc1.txt | 原创文本 |
| 文本 | doc1\_same.txt | 与doc1.txt相同 |
| 文本 | doc2.txt | 与doc1.txt相似 |
| 文本 | Doc3.txt | 不同内容文本 |
| 图像 | MonaLisa\_1.jpg | 原创图像 |
| 图像 | MonaLisa\_2.jpg | 相似图像 |
| 图像 | Other.jpg | 另一原创图片 |
| 音频 | heaven.mp3 | 原创音频 |
| 音频 | Heaven\_first\_half.mp3 | 原创音频前半段 |
| 音频 | heaven\_=second\_half.mp3 | 原创音频后半段 |
| 音频 | Heaven\_small.mp3 | 相似音频 |

**（3）运行Linux下的版权保护区块链系统**

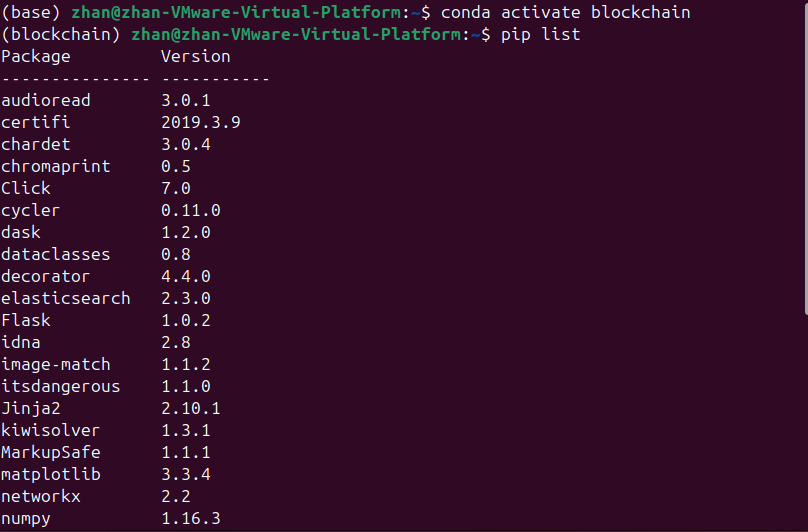
1.首先打开Linux虚拟机，并且从云平台下载实验的源代码，在Linux环境下使用Pycharm打开，如下图所示。



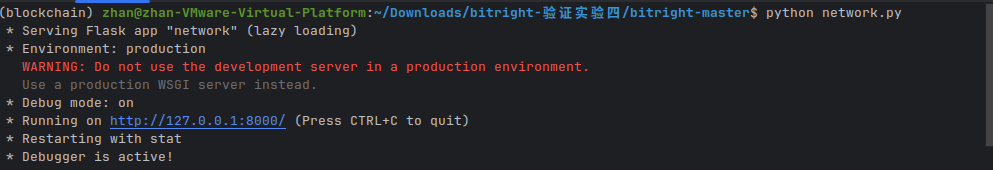
2.由于运行source ./install.sh时频繁报错，所以打算使用anaconda配置虚拟环境，下载anaconda，并且创建一个新的虚拟环境blockchain。



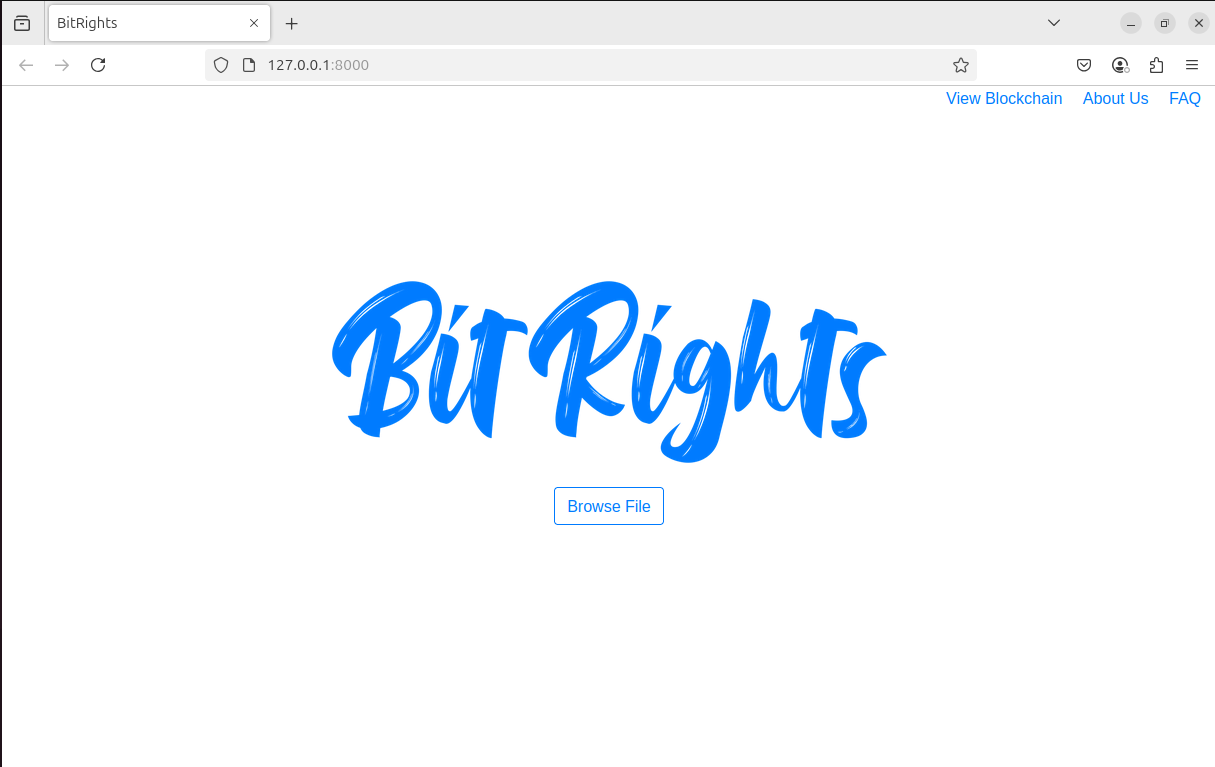
3.安装requirement.txt中要求安装的包，如下图所示。

****

4.使用python network.py 运行该版权保护区块链系统，如下图所示。

****

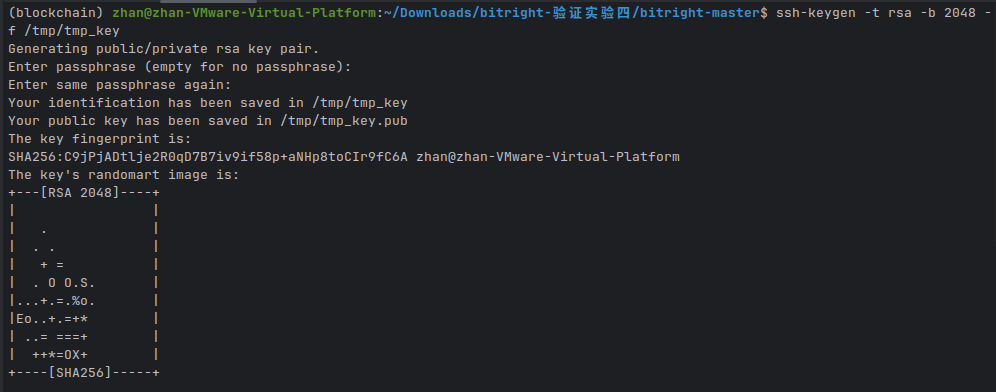
5.点开网址http://127.0.0.1:8000/，即打可开版权保护区块链系统，如下图所示。

****

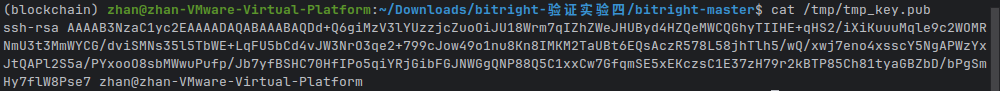
**（4）用版权保护区块链系统对实验测试数据提取版权信息进行不重复的上链。**

**1.生成SSH秘钥对**

①在上传文件是需要SSH密钥对的公钥，所以我们先生成一对SSH密钥对，使用以下命令即可。ssh-keygen -t rsa -b 2048 -f tmp/tmp\_key

****

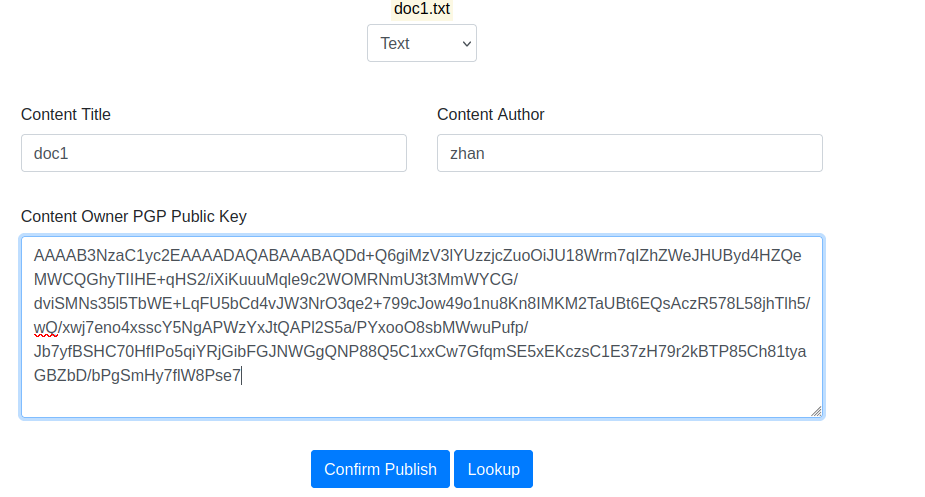
②生成完毕之后，还需要查看公钥，使用以下命令即可。Cat /tmp/tmp\_key.pub

****

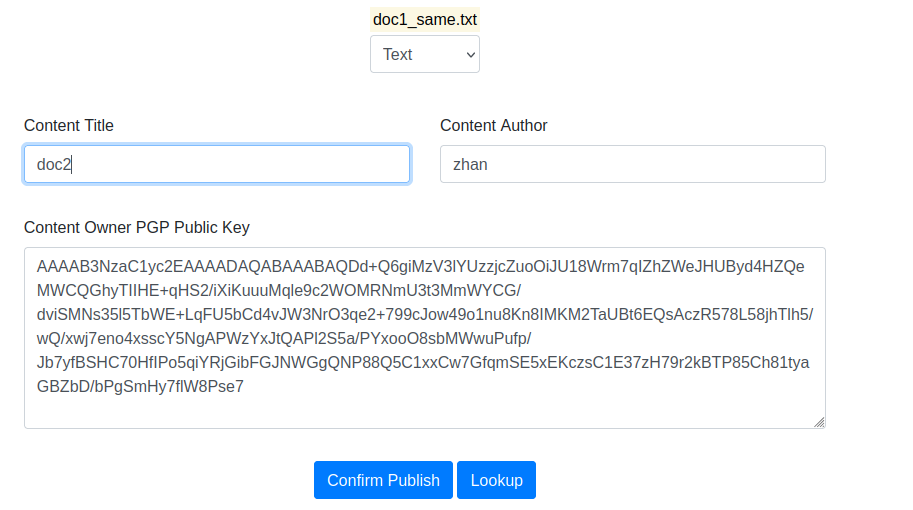
**至此，SSH秘钥对生成完毕，并且已经获取了公钥。**

**2.对文本文件进行测试。**

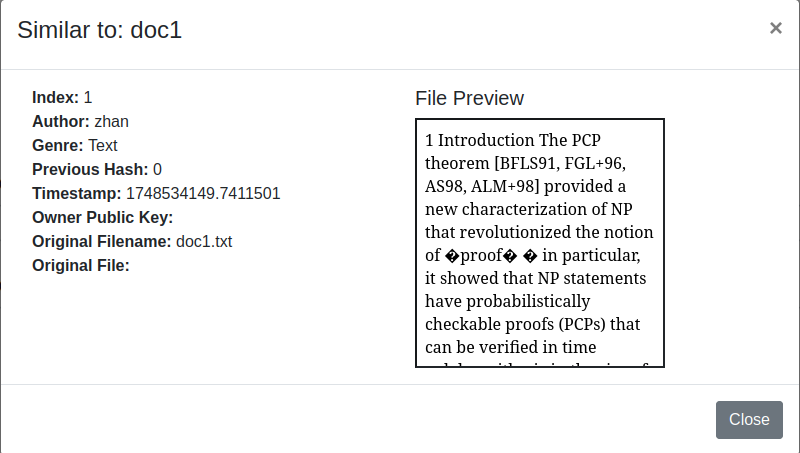
①选择doc1.txt进行上传，Cotent Title,Content Author都可以自选，输入第一步中得知的公钥，点击Confirm Publish，即可上传成功。

****

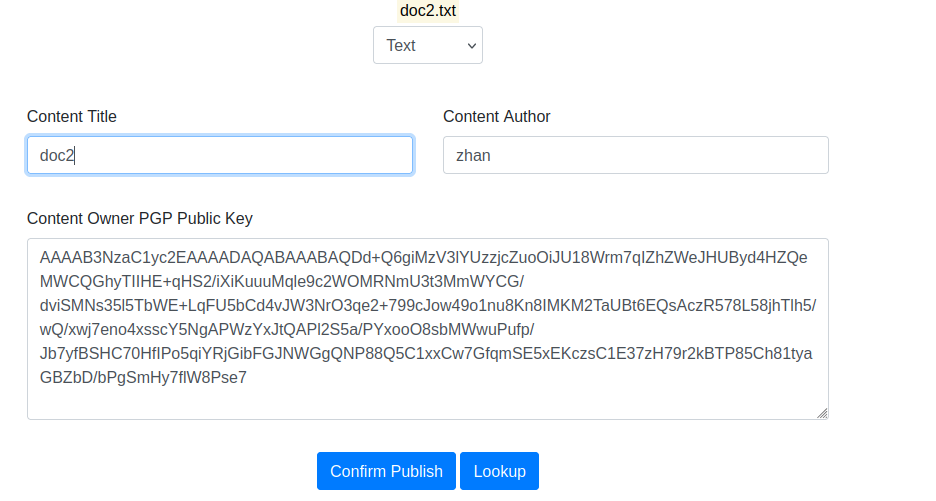
②选择doc1\_same.txt进行上传，该文件与doc1.txt完全相同。



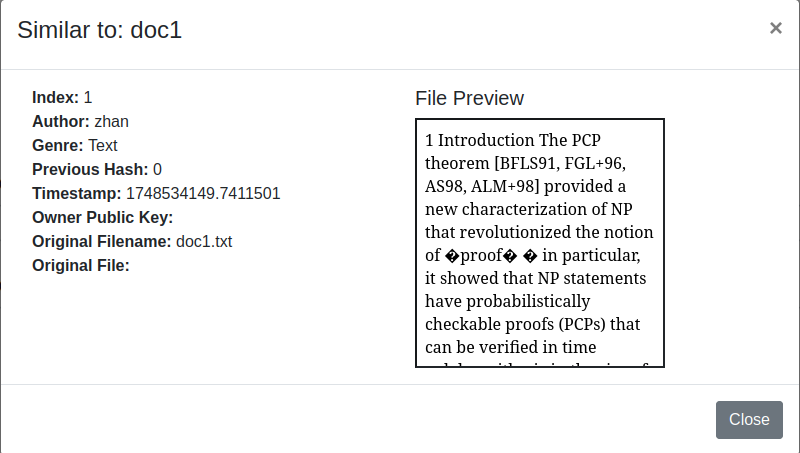
③我们会发现文件无法上传，因为和之前的文本相似，具体信息如下图所示。

****

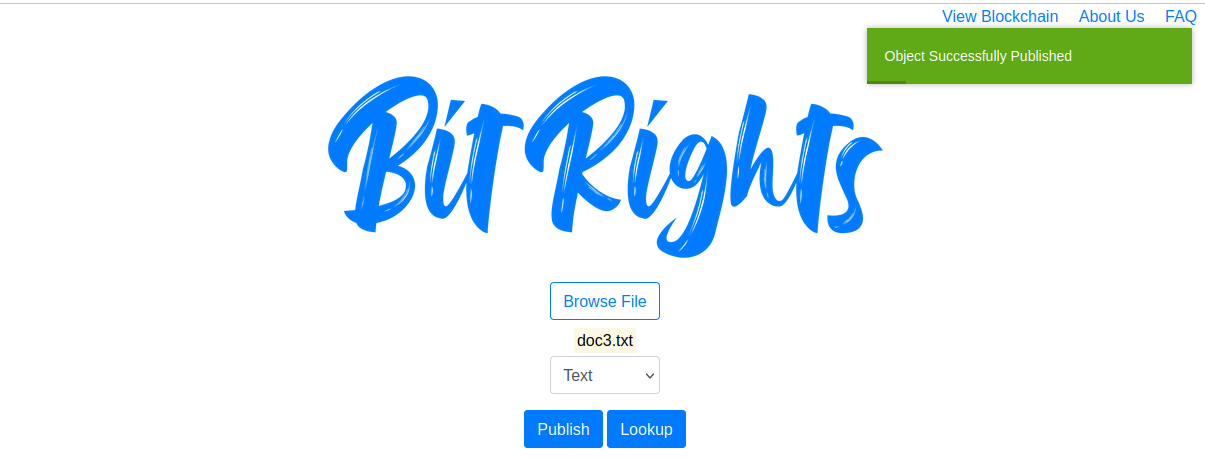
④选择doc2.txt进行上传，该文件与doc1.txt高度相似。

****

⑤我们会发现文件无法上传，因为和之前的文本相似，具体信息如下图所示。

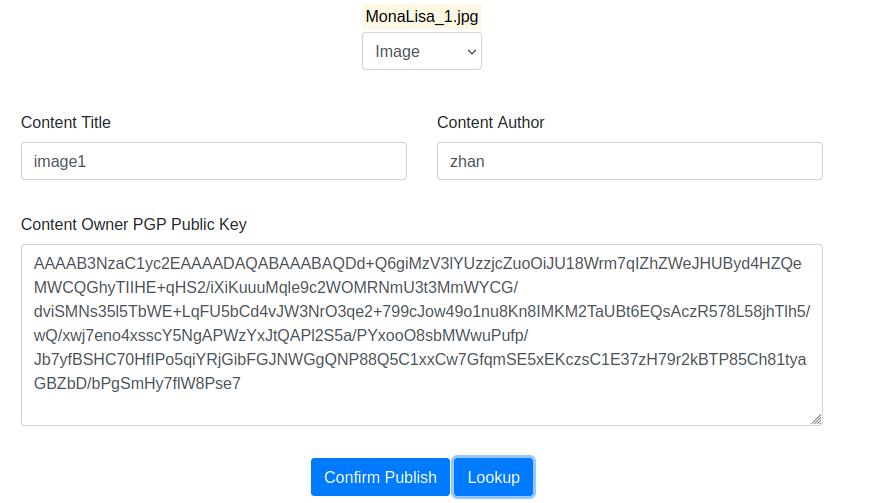
****

⑥选择doc3.txt进行上传，该文件与doc1.txt完全不同，我们发现该文本文件可以成功上传。

****

**3.对图像文件进行测试。**

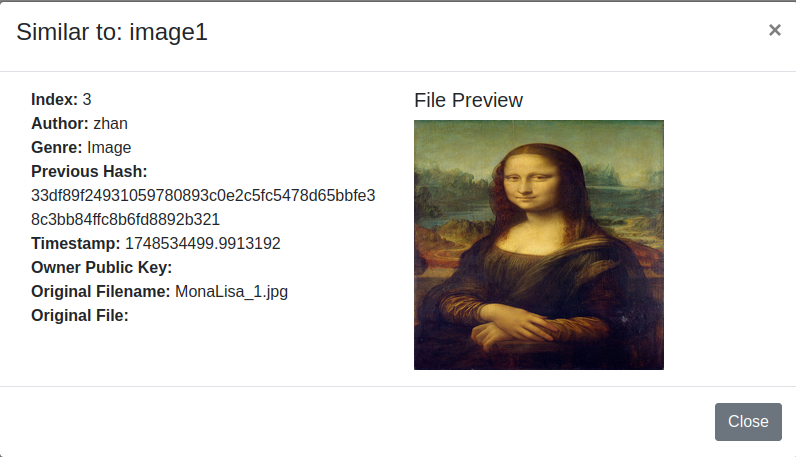
①选择MonaLisa\_1.jpg进行上传，该文件为原创图像，点击Confirm Publish，即可上传成功。

****

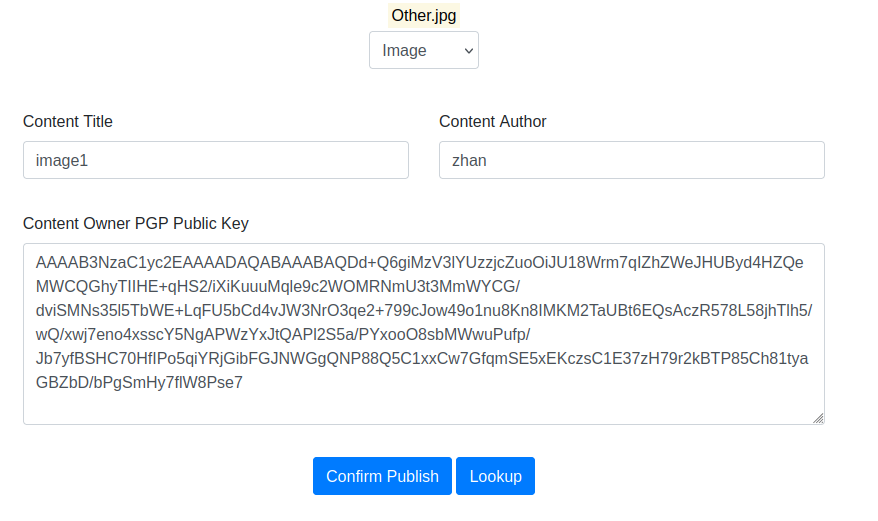
我们可以发现该图像文件已经成功上传。

****

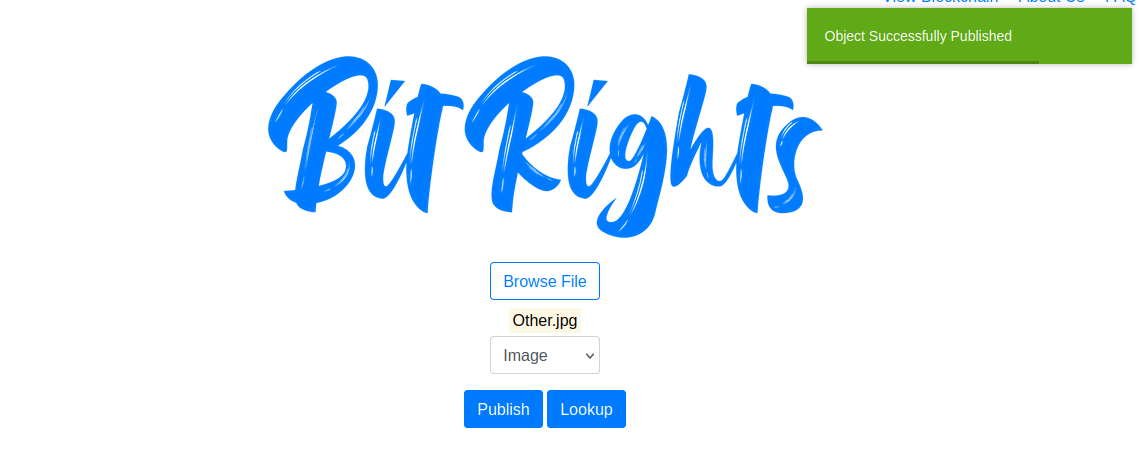
②选择MonaLisa\_2.jpg进行上传，该图像文件与MonaLisa\_1.jpg高度相似，点击上传时发现无法上传，点击Lookup发现错误信息，与MonaLisa\_1,jpg高度相似。

****

③选择Other.jpg进行上传，该图像文件和MonaLisa\_1.jpg完全不同。

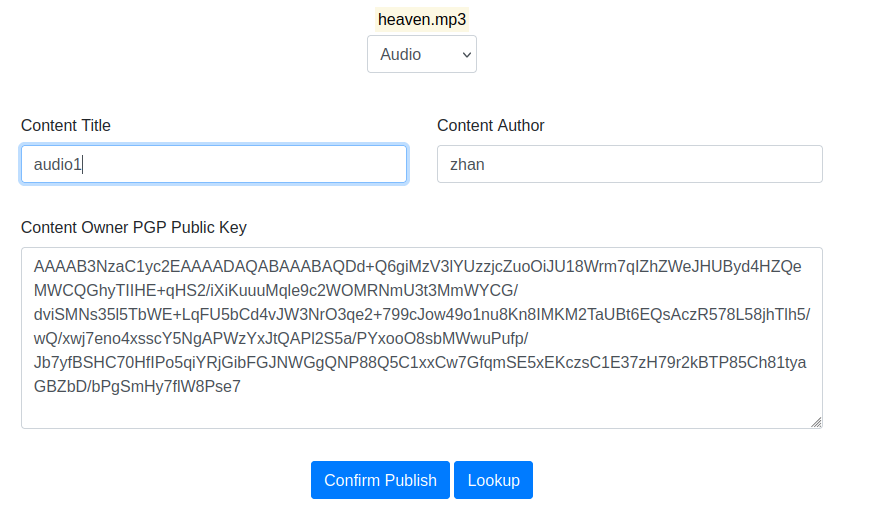
****

④我们可以发现该图像文件已经成功上传。

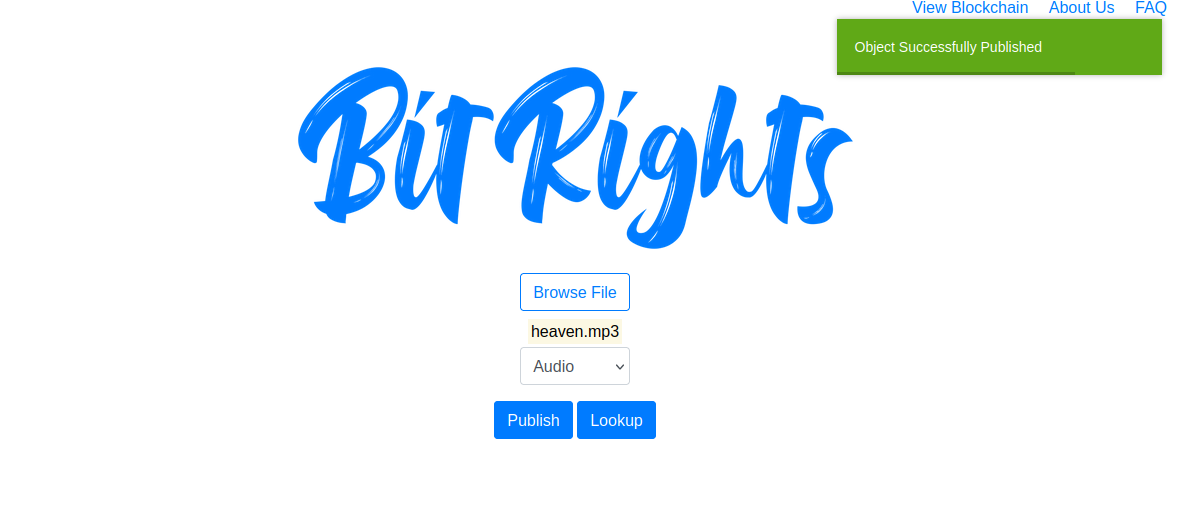
****

**4.对音频文件进行测试。**

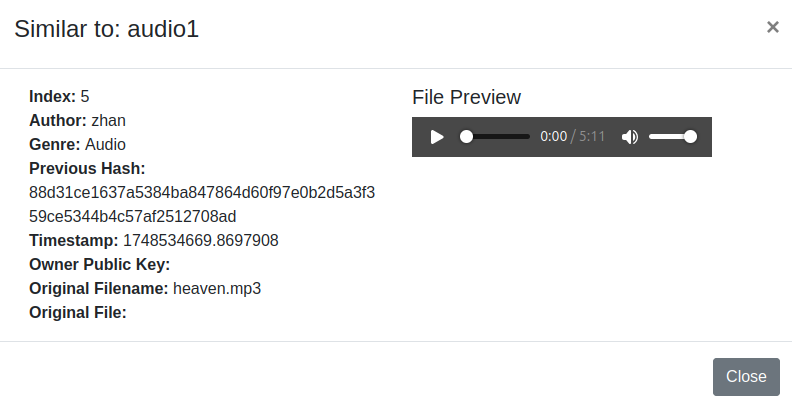
①选择heaven.mp3进行上传，该文件为原创音频，点击Confirm Publish，即可上传成功。

****

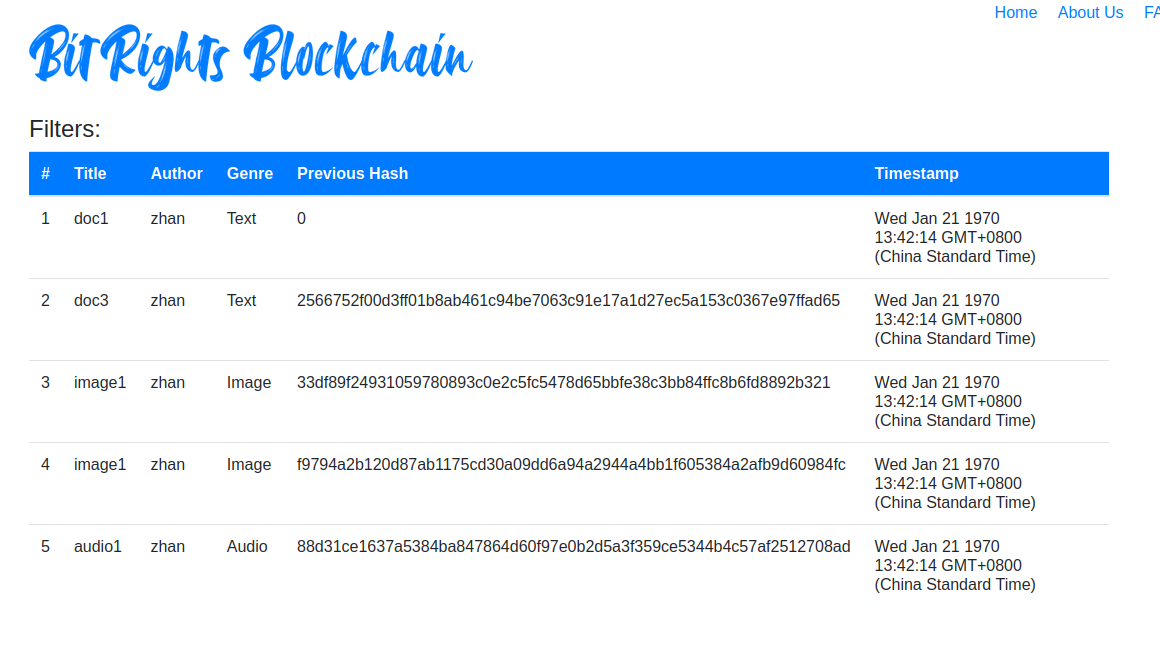
②我们可以发现该音频文件可以成功上传。

****

③接下来我们尝试上传media下其他的音频文件，均显示与heaven.mp3高度相似，无法上链，（此处就不一一演示了）。

****

④最终所有上传的文件均可以在View Blockchain中查看，点击可以看到上传的五个文件。

****

**（5）实验结果及分析**

本次实验围绕BitRights区块链版权保护系统，对文本、图像、音频三类数据进行了版权登记与查重测试。系统通过区块链技术结合多模态查重算法，成功实现了原创性验证与版权上链目标。实验结果如下：

1.文本数据测试分析

* 上传原创文本doc1.txt，系统成功完成上链，生成新块。
* 上传与其完全相同的doc1\_same.txt，系统通过TLSH算法检测到两者文本哈希距离为0，提示“相似对象已存在”，禁止重复上链。
* 上传部分相似文本doc2.txt，TLSH距离仍小于设定阈值（<100），系统依旧判定为非原创，拦截上链。
* 上传与原文本内容差异较大的doc3.txt，系统判断为原创内容，允许发布并上链。

该过程表明，TLSH算法对文本的微小改动具有较强的敏感性，能够有效避免文字抄袭内容的重复注册。

2.图像数据测试分析

* 上传MonaLisa\_1.jpg图像，系统识别为原创图像并成功上链。
* 再次上传相似图像MonaLisa\_2.jpg，系统基于感知哈希提取图像特征签名并计算距离，结果表明图像过于相似，系统拒绝上链。
* 上传不同图像Other.jpg，经比对相似度低于阈值，系统认定为独立图像，完成上链。

该过程表明，image-match模块对图像相似度识别效果良好，适合检测伪造、轻度修改的图片作品。

3.音频数据测试分析

* 上传heaven.mp3为原创音频文件，系统成功提取其指纹信息并完成版权登记。
* 再次上传对该音频的截取版本、压缩版本（如Heaven\_first\_half.mp3,Heaven\_small.mp3），系统通过AcoustID+Chromaprint比对其音频指纹，相似度高于设定阈值（>0.9），全部识别为重复作品并阻止上链。

该过程表明，音频指纹识别算法在版权检测中的有效性，尤其适用于音乐类作品的防伪与重复发布控制。

4.区块链运行与数据管理分析

* 每一个成功的上传行为，系统都会创建一笔新交易，并打包进新区块。
* 所有区块结构清晰、字段完整，包含内容标题、作者、公钥、时间戳、前一区块哈希等信息。
* 系统成功生成了一个长度为5的有效区块链，分别对应2个文本、2个图像、1个音频文件。

该部分验证了区块链结构的基本可用性和不可篡改性，能够作为数字版权登记和防篡改数据存储的有效工具。

**五、实验总结**

通过本次实验，我全面掌握了版权保护区块链系统的架构、功能和运行机制。在实际部署与测试过程中，有以下几点总结与体会：

（1）掌握了区块链核心概念

实验系统通过构建区块（Block）和链（Blockchain），将版权登记行为转化为不可篡改的数据记录，从技术层面实现了作品所有权的“可信登记”。

（2）深入理解了跨模态查重算法的原理与实现

系统分别采用TLSH、AcoustID、PerceptualHash等技术实现对文本、音频和图像的原创性识别，显著提升了版权保护的精度与自动化程度。

（3）具备了一套完整的区块链版权登记流程实践能力

包括文件上传、原创性验证、交易构建、区块打包、链式结构存储等关键环节。

（4）验证了系统的可扩展性和稳定性

Web界面友好，系统部署于本地服务器，上传行为反馈及时，且链条结构清晰持久化良好，具备一定的生产级原型雏形。

（5）强化了开发与调试技能

在Linux+Flask环境中完成了模块安装、依赖管理、虚拟环境构建、文件读写权限处理等多个技术细节的实践，提升了系统集成与问题定位的能力。

**六、实验中遇到的问题及改正的方法**

（1）使用source ./install.sh安装相关库时，一直发生报错，所以后面舍弃该种方法，使用anaconoda配置虚拟环境，虽然过程中也会遇到一些包无法安装的情况，但是后续也是成功解决，配置好了区块链系统的运行环境。

