// SPDX-License-Identifier: MIT

pragma solidity ^0.8.5;

contract FileStorage {

    uint256 public uploadLimit;

    address public owner;

    bytes32 public lastFileID; // Переменная состояния для хранения последнего загруженного fileId

    mapping(bytes32 => bytes) private files; // Маппинг для хранения файлов по fileId

    event FileUploaded(bytes32 indexed fileId, address indexed uploader, uint256 fileSize);

    constructor() {

        owner = msg.sender;

        uploadLimit = 1000000; // Установим лимит газа в 1 миллион (по умолчанию)

    }

    function setUploadLimit(uint256 \_newLimit) public {

        require(msg.sender == owner, "Only owner can set upload limit");

        uploadLimit = \_newLimit;

    }

    function uploadFile(bytes32 fileId, bytes memory fileData) public {

        require(fileData.length <= uploadLimit, "File size exceeds upload limit");

        // Сохраняем fileId и fileData в маппинге files

        files[fileId] = fileData;

        // Обновляем переменную lastFileID

        lastFileID = fileId;

        emit FileUploaded(fileId, msg.sender, fileData.length);

    }

    function getFileSize(bytes32 fileId) public view returns (uint256) {

        return files[fileId].length; // Возвращаем размер файла по его fileId

    }

    function getFile(bytes32 fileId) public view returns (bytes memory) {

        return files[fileId]; // Возвращаем содержимое файла по его fileId

    }

    function getLastFileID() public view returns (bytes32) {

        return lastFileID; // Возвращаем последний сохраненный fileId

    }

}