**Communication Between Two Nodes: Sharing Bin Files and Decoding Back to the Original One**

This project demonstrates a communication process between two nodes using Flask web servers. Each node is equipped with a Flask server for receiving binary files, and they use a key to encode and decode the data.

**Requirements**

- Python 3.x

- Flask

- Flask-CORS

- FFmpeg

**Installation**

1. Install Python 3.x from [https://www.python.org/downloads/](https://www.python.org/downloads/).

2. Install Flask and Flask-CORS using the following command:

```bash

pip install Flask Flask-CORS

```

3. Install FFmpeg from [https://ffmpeg.org/download.html](https://ffmpeg.org/download.html).

**Usage**

**Node 1: Sending Encoded Bin Files**

- Run the Flask server on Node 1 using the provided script (`node1\_server.py`).

- Ensure the server is running on `http://localhost:5001/`.

**Node 2: Receiving and Decoding Bin Files**

- Run the Flask server on Node 2 using the provided script (`node2\_server.py`).

- Ensure the server is running on `http://localhost:5000/`.

**Endpoints**

**Node 1:**

- POST `/` endpoint:Uploads a binary file, decodes it, and deletes the binary file after decoding.

**Node 2:**

- POST `/` endpoint:\*\* Uploads an encoded binary file to Node 1 for further processing.

**Key Handling**

- The key used for encoding and decoding is set to `66`. You can adjust this key as needed in the script.

**Example Usage**

1. Start both Flask servers on Node 1 and Node 2.

2. Send a binary file to Node 1 using the `/` endpoint.

3. Node 1 will decode the file, and the decoded binary file will be sent to Node 2 using the `/endpoint-two` endpoint.

**Note:**

There are already two hardcoded bin files (`example\_bin\_file1.bin` and `example\_bin\_file2.bin`) used for demonstration purposes. You can replace these with your own binary files.