**Project Proposal: Implementation of a Peer-to-Peer (P2P) File Sharing System**

**Group Members**

* **Name:** [Aun Ali Kazi]
* **ID:** [22k-4029]
* **Name:** [Uzair ahmed]
* **ID:** [22K-4070]

**Proposed Project Description**

The P2P File Sharing System is a decentralized application that enables users to share files directly with each other without relying on a central server. The system uses socket programming for direct file transfers and Django for the web interface and backend management. Files are encrypted during transfer to ensure security.

How It Works

* Users enter the IP address and port number of the peer they want to connect to
* Files are shared directly between peers using socket connections
* Django provides a user-friendly web interface for initiating file transfers and viewing shared files
* Files are encrypted using Fernet symmetric encryption before transmission and decrypted upon receipt
* A separate receiver server runs alongside the Django application to handle incoming file transfers

**Functional Features**

* **Direct File Sharing** – Users can send and receive files without a central server by specifying the receiver's IP and port.
* **Django-based Backend Management** – Manages file-sharing requests and provides a web interface for the system.
* **Efficient File Transfer** – Utilizes socket-based communication with chunked file reading/writing for smooth data transmission.
* **Basic Frontend Interface** – Minimal HTML, CSS, and JavaScript interface for initiating transfers and viewing shared files.
* **Security Measures** – Uses Fernet symmetric encryption to secure file transfers with proper key handling.
* **File Preview** – Allows users to view the contents of text files directly in the browser.

**Plan of Work (5 Weeks)**

**Week 9:**  
• Research P2P file-sharing concepts and socket programming.  
• Set up the project environment and tools (Python, Django, Sockets, HTML, CSS, JS).  
• Design the system architecture and workflow.

**Week 10:**  
• Implement basic socket communication for file transfers.  
• Set up the Django project structure and basic views.  
• Create the receiver server to handle incoming files.

**Week 11:**  
• Implement file-sharing with direct peer-to-peer transfer.  
• Develop Django logic to handle transfer requests.  
• Begin developing a simple frontend interface.

**Week 12:**  
• Add encryption for secure file transfers.  
• Optimize file transfers using chunked reading/writing.  
• Finalize and polish the frontend.

**Week 13:**  
• Perform final debugging and testing.  
• Enhance UI with improved styling and feedback.  
• Complete documentation and prepare the project presentation.

**Team Responsibilities**

* **Uzair:** Django backend management, frontend development, UI design.
* **Aun:** File transfer implementation, socket programming, and security features.

**References**

1. Relevant online tutorials on decentralized systems.

This project aims to provide an efficient and decentralized method for users to share files directly without the need for a centralized server, leveraging Python, Django, and socket programming for seamless operation.