SYNOPSIS
Report on

ShareTo

by

Vishal Sharma 2300290140205
Yatharth Kaushik 230290140212
Tapasya 230290140193
Shivam Singh 230290140171

Session:2023-2025 (III Semester)

Under the supervision of

Ms. Annu Yadav (Assistant Professor)

KIET Group of Institutions, Delhi-NCR, Ghaziabad

DEPARTMENT OF COMPUTER APPLICATIONS
KIET GROUP OF INSTITUTIONS, DELHI-NCR, GHAZIABAD-201206
(2023 - 2025)

ABSTRACT

ShareTo is a desktop application designed to enable seamless file and data sharing between two PCs over a local or remote network. The application leverages technologies like Socket.IO or WebRTC to establish a peer-to-peer connection, ensuring fast and secure data transfer. By simplifying the process of sharing files across systems, ShareTo aims to enhance collaboration in both personal and professional settings. This application supports various file types, provides an intuitive user interface, and ensures encryption during file transfer to protect sensitive data. With minimal configuration required, ShareTo offers a user-friendly solution to a common problem faced in networking environments.

Keywords: File Sharing, Peer-to-Peer, Socket.IO, WebRTC, Encryption.

TABLE OF CONTENTS

INTRODUCTION

File sharing has become an essential task in today's digital world, whether for personal, academic, or professional purposes. Users often need to transfer files between multiple systems, yet existing solutions can be cumbersome, involving email attachments, cloud-based file transfers, or the use of external devices like USB drives. These methods, while effective, have their limitations in terms of speed, ease of use, security, and dependency on internet connectivity (in the case of cloud services).

The rise of peer-to-peer (P2P) file-sharing applications offers a more streamlined approach. P2P systems allow direct sharing of files between two computers without needing intermediary storage, significantly reducing transfer times and eliminating third-party intervention. However, existing P2P applications often require complex setups, involve risks like unauthorized data access, or lack security protocols.

ShareTo, as a desktop application, aims to address these challenges by providing a fast, secure, and user-friendly platform for transferring files between two PCs. ShareTo can work over both local networks and the internet, allowing users to exchange files without the need for external media or complex configurations.

LITERATURE REVIEW

Various file-sharing solutions have existed over the years, each with its advantages and drawbacks:

1. **Email Attachments**: One of the earliest methods of file sharing is through email. While convenient for smaller files, it is inefficient for large file transfers, often having file size limitations, and the process is cumbersome for frequent exchanges.
2. **Cloud Storage Services (e.g., Google Drive, Dropbox)**: These platforms allow users to upload files to the cloud and share links to others. While this is useful for remote sharing and backup, it requires an internet connection and often involves complex sharing permissions and file access controls. Furthermore, cloud services can be slow depending on the network speed and have inherent privacy concerns.
3. **External Devices (USB, HDD)**: Transferring data through physical devices like USB flash drives is fast and secure when both systems are in the same location. However, this method becomes impractical when users are working remotely, and there's a risk of losing or damaging the storage device.
4. **Peer-to-Peer (P2P) Sharing (BitTorrent, etc.)**: P2P applications allow direct file transfers between two or more devices. These platforms offer speed advantages and work well over large networks. However, they often face issues related to security, privacy, and copyright infringement due to a lack of secure encryption.

Despite the array of options, many users still face issues with file size limitations, network dependencies, security risks, and difficulties in setup. **ShareTo** fills this gap by offering a simplified solution with an emphasis on security, speed, and ease of use.

PROJECT OBJECTIVE

The objective of the **ShareTo** project is to develop a desktop application that allows for quick, secure, and straightforward file sharing between two PCs, either over a local network or the internet. The application will prioritize:

1. **Ease of Use**: Minimal configuration should be required, making the platform accessible for all types of users.
2. **Security**: The application will utilize encryption to ensure secure data transfer, preventing unauthorized access to the files.
3. **Efficiency**: The solution should provide real-time transfer of data, optimizing the speed depending on the network bandwidth.
4. **Cross-Platform Support**: ShareTo will work across various operating systems, ensuring compatibility between different types of PCs.

The aim is to create a tool that reduces the complexities associated with file sharing, ensuring that users can quickly and securely share their files without reliance on external cloud services or physical media.

PROJECT FLOW / RESEARCH METHODOLOGY

1. **System Architecture**:

   The application will be based on a peer-to-peer (P2P) architecture. This architecture allows the direct transfer of files between two PCs without requiring an intermediary server. A connection will be established between the two computers, and files will be sent over this connection.

   - **Frontend**: A simple, intuitive graphical user interface (GUI) developed using technologies like Electron or Qt, allowing users to easily drag-and-drop files, select recipient PCs, and initiate transfers.
   - **Backend**: The core file transfer mechanism will be developed using technologies such as Socket.IO or WebRTC. Socket.IO provides a real-time bidirectional communication layer, whereas WebRTC offers a more direct peer-to-peer data channel, ensuring fast data transfer and security through encryption protocols.
   - **Security**: All file transfers will be encrypted using modern encryption standards such as AES (Advanced Encryption Standard), ensuring the safety and privacy of the data during transit.

2. **Technologies Used**:

   - **Socket.IO**: A real-time engine used for sending data in both directions, which is ideal for peer-to-peer file transfers.
   - **WebRTC**: An alternative technology designed for establishing a direct connection between two clients, bypassing any central server, thus offering faster and more secure transfers.

- **Encryption**: AES will be used for encrypting the files before transmission. This ensures that even if the transfer is intercepted, the data remains secure.

3. **Development Process**:
    - **Phase 1**: Requirements gathering, selecting the appropriate technology stack (Socket.IO vs WebRTC), and designing the user interface.
    - **Phase 2**: Backend development – setting up the peer-to-peer connections and implementing the file transfer logic.
    - **Phase 3**: Integrating security features and ensuring encrypted transfers.
    - **Phase 4**: Testing and optimization for network efficiency and user experience.

PROJECT OUTCOME

The expected outcome of the **ShareTo** project is a fully functional desktop application that enables secure, real-time file sharing between two PCs. The application should:

- Support multiple file types and sizes.
- Work on local networks as well as over the internet.
- Offer secure, encrypted file transfers.
- Provide a simple, user-friendly interface requiring minimal setup.

Successful implementation will result in a tool that significantly enhances file-sharing efficiency in various environments, from personal use to professional collaboration.

PROPOSED TIME DURATION

The estimated time duration for the development and testing of **ShareTo** is as follows:

1. **Requirement Analysis and Planning**: 2 weeks
2. **Design and Technology Selection**: 1 week
3. **Frontend and Backend Development**: 5 weeks
4. **Security Implementation (Encryption)**: 2 weeks
5. **Testing and Bug Fixing**: 3 weeks
6. **Final Deployment and Documentation**: 1 week

Total Duration: **14 weeks** (3.5 months)

This timeline accounts for both development and thorough testing to ensure the application meets performance, security, and user experience expectations.