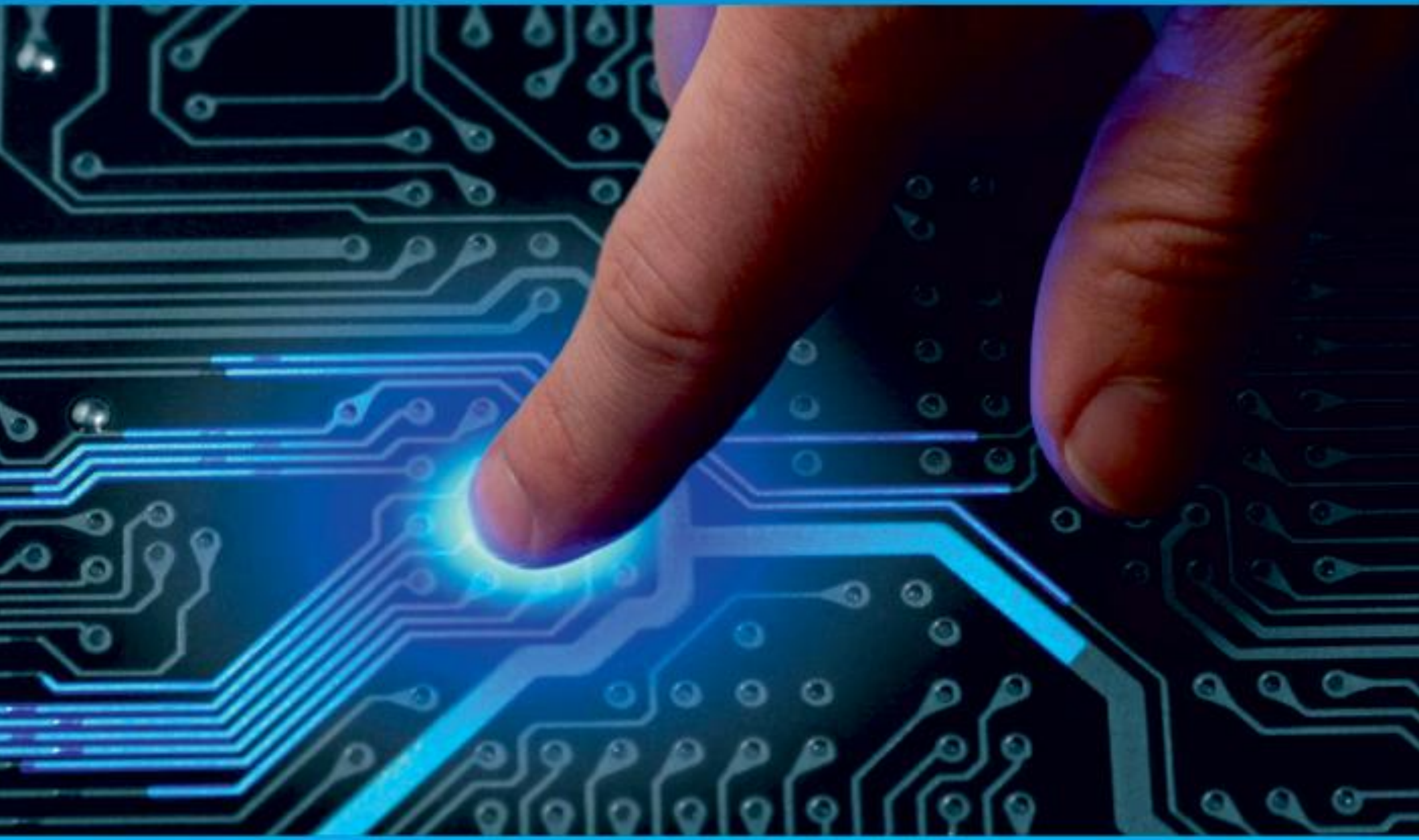
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 11, Issue 4, April 2023

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.379

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104117 |

# Cross Platform File Transfer (Craft)

**S.P. Bansu, Aditya Kesarwani, Anujeet Prajapati, Rohan Mestry, Omkar Zagade**

Dept. of Computer Engineering, A C Patil College of Engineering, University of Mumbai, Navi Mumbai, India

**ABSTRACT:** The demand for fast and secure peer-to-peer file transfer systems has increased with the rise of digital content. In this paper, we propose a web application that leverages the advantages of peer-to-peer networks to facilitate efficient and secure file transfers between users. The web application is designed to establish direct peer-to-peer connections between users using the WebRTC (Web Real-Time Communications) API, eliminating the need for intermediate servers or cloud-based storage. The system employs end-to-end encryption to ensure the confidentiality and integrity of transferred files. To ensure scalability and fault tolerance, the proposed system architecture is designed to accommodate a growing user base while maintaining seamless operation. The user interface is user-friendly, providing features such as drag-and-drop file uploads and real-time progress updates.

## I. Introduction

This research paper explores the development and implementation of a peer-to-peer file sharing web app using WebRTC technology. WebRTC provides direct and efficient communication between peers without the need for a central server or intermediary, making it a powerful solution for real-time communication and data sharing over the internet. We will discuss the benefits of using WebRTC for peer-to-peer file sharing, evaluate the app's performance and usability, and suggest areas for further research and development. The paper aims to provide a comprehensive overview of the potential of WebRTC for creating decentralized, user-controlled communication networks that prioritize privacy and security.

## II. Literature Survey

Survey of Existing System: To fully comprehend and evaluate the potential impact of Peer-to-Peer networking, a clear and comprehensive definition of the technology is needed. The lack of consensus on the terminology used in this field has hindered meaningful discussions and publication. While progress has been made, such as the development of dynamic routing concepts, there are still obstacles to overcome before Peer-to-Peer networking can achieve its full potential as a third-generation internet technology[1].

BitTorrent is no longer exclusive to desktop computer users, as mobile users can now also take advantage of distributed file-transfer with SymTorrent and MobTorrent. This paper provides an overview of the available mobile BitTorrent technologies and compares their performance. The native version performs better in most cases, but it is limited to Symbian-based smartphones. While MobTorrent has some shortcomings such as compatibility issues and socket-related problems, most of these are related to Java rather than our implementation. We anticipate that future versions of the Java virtual machine will address these issues[2].

Numerous academic researches have discussed Peer-to-Peer (P2P) file sharing systems, including popular applications such as Gnutella, Napster, BitTorrent, and SymTorrent for both computers and mobile devices. File sharing generates significant network traffic, so technologies have been developed to reduce it and simplify file searches. This paper presents a P2P file sharing system between mobile devices that utilizes Bluetooth as a communication protocol. The application permits sharing and publishing files between J2ME and MIDP-enabled devices over Bluetooth, enabling users to search for and download specific file types, such as music, pictures, text, and programs. Future development may include additional features and the capacity to operate on various mobile operating systems[3].

The use of Socket Programming in Peer-to-Peer (P2P) systems requires a client/server application. The program consists of a concurrent server program, PEER-A, that waits for a connection from a client program, PEER-B, running on a different machine. Once the connection is established, PEER-A sends a database containing a list of files it's willing to share with PEER-B. PEER-B can then select a file to view from PEER-A, and a history table records the

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104117 |

imported file, its source, and the time it took to transfer. Each peer can add new files to the database, which requires PEER-A to establish a new connection with PEER-B to update the database. Finally, after file transfer, the socket connections must be closed[4].

The study focuses on establishing a connection between two peers using a client-server architecture and sharing files between them. This knowledge can be applied to building an interface and architecture that can be used for creating a serverless application that works on multiple platforms. The ability to share files between peers without the need for a centralized server can be a significant advantage in terms of privacy and security, and it also allows for more efficient use of network resources. By utilizing the concepts outlined in the research, developers can design a robust and scalable serverless application that can meet the needs of users across different platforms.

## III. PROPOSED SYSTEM

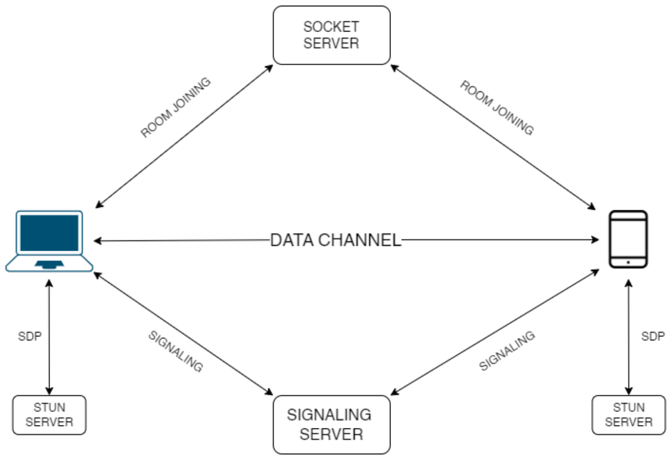
### A. *Design Considerations:*

After analyzing existing file transferring applications, we have identified several issues including registration/sign-in, software compatibility, data logging, software installation, and privacy concerns. To address these issues, we propose the development of a web-based file transferring system that eliminates the need for sign-in/registration and software installation by using a browser-based project. Additionally, we plan to incorporate WebRTC (Web Real-Time Communication), which is a free and open-source project that provides real-time communication between web browsers and mobile applications via APIs. By using WebRTC, we can also address concerns related to data logging and privacy.

### B. *Architecture:*

*III.B.1* Connecting to Signalling Server

The initial stage of the file transfer process involves connecting to the signalling server, which is achieved via an HTTP request. This HTTP request sends a signal to the hosted signalling server to initiate the connection.



*III.B.2* Establishing a p2p channel

In order to create a peer-to-peer (p2p) channel between two clients for file transfer, the web application utilizes the signaling server for signaling. After the two clients are brought into the same room, they can exchange connection details such as session description protocol (SDP) and ICE candidates. These connection details are essential for establishing the p2p channel between the sender and receiver.

SDP — A Session Description Protocol, which is an object containing details about the session connection, such as the codec, media type, address, and audio/video information. During the file transfer, both sender and receiver exchange their SDPs to establish a connection. One of the peers sends an SDP offer, and the other responds with an SDP answer, allowing them to understand how to connect with each other[7].

ICE Candidates —An ICE candidate is a combination of a public IP address and port number that can serve as a potential target for receiving data. Usually, each user has multiple ICE candidates that are obtained by sending requests to a STUN server.

*III.B.3* Chunking of data

After establishing a connection, the files are transferred in chunks using the slice function. This function creates a new array buffer containing the bytes in the specified range, which is considered as a chunk. The chunk is removed from the existing buffer, and this process is repeated until the entire file is chunked. However, it is unclear what should be the ideal size for each chunk since different browsers have different WebRTC implementations, and newer browsers may not be compatible with older ones. After some research, it has been determined that the maximum safe limit for each chunk is 16KB. Chunking provides several benefits such as the ability to support large file sizes, better monitoring of the data transfer, and detection of incomplete file transfers[5].

Benefits of chunking:
Large file size support
A better way to interpret the amount of data transferred • Detect incomplete file transfers

*III.B.4* Reassembling of data

The chunked files are combined into a blob at the receiver side. The blob is the file which is sent for the sender. The file can then be downloaded by the receiver.

C. *Requirement Analysis:*

Hardware and software requirements for the system are outlined as follows. The system requires a minimum of 512 MB of RAM and 150 MB of disk space. The CPU must be an Intel Pentium 4 or higher. In terms of software, the system is compatible with Microsoft Edge 12 or later, Google Chrome 28 or later, Mozilla Firefox 22 or later, Safari 11 or later, Opera 18 or later, Vivaldi 1.9 or later, and Brave web browsers.

## IV. RESULTS

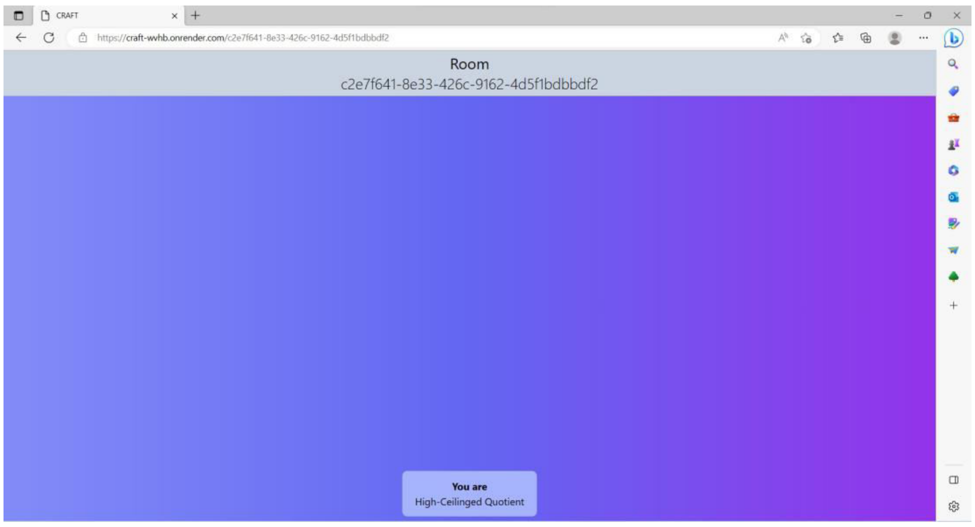

CRAFT
https://craft-wvhb.onrender.com/c2e7f641-8e33-426c-9162-4d5f1bdbbdf2
Room
c2e7f641-8e33-426c-9162-4d5f1bdbbdf2
You are
High-Ceilinged Quotient

Figure 1 : Creating a room

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 4, April 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1104117 |

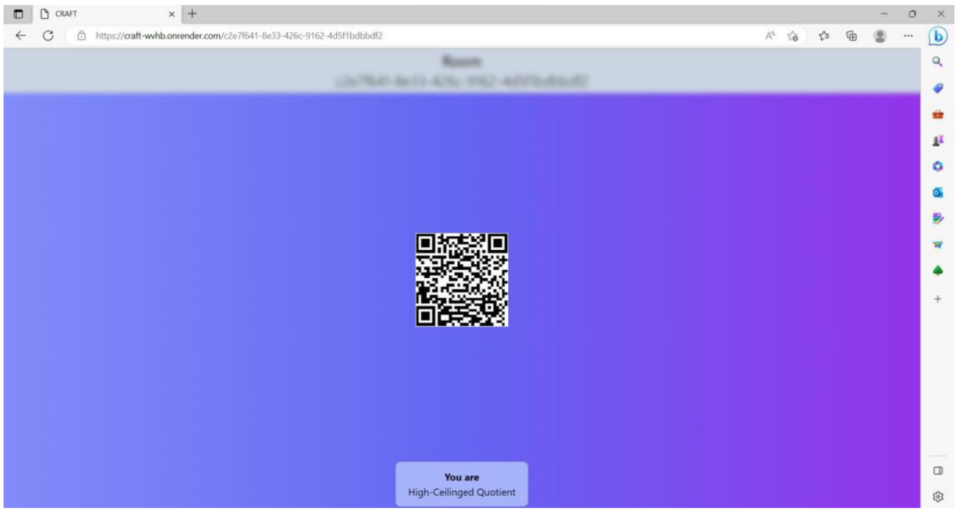
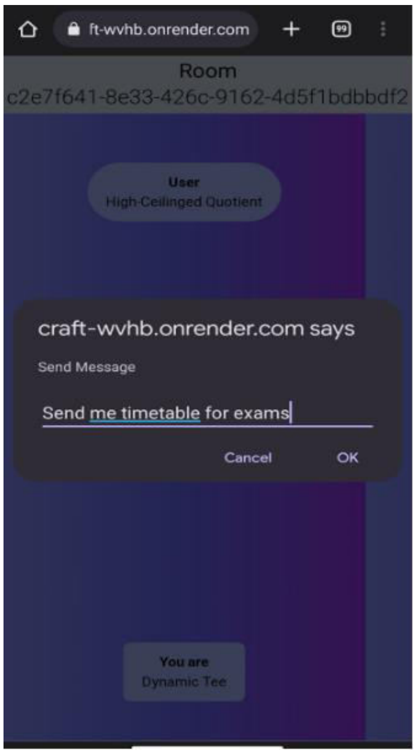

CRAFT
https://craft-wvhb.onrender.com/c2e7f641-8e33-426c-9162-4d5f1bdbbdf2
You are
High-Ceilinged Quotient

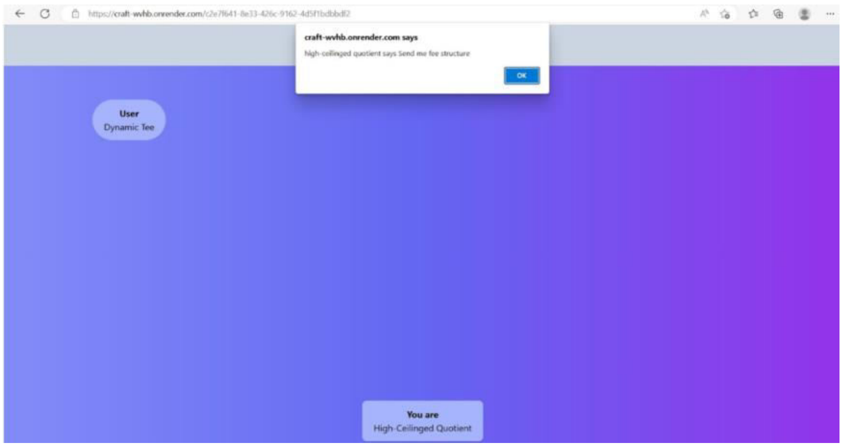
Figure 2: Sharing a QR code to join in room



craft-wvhb.onrender.com says

Send Message

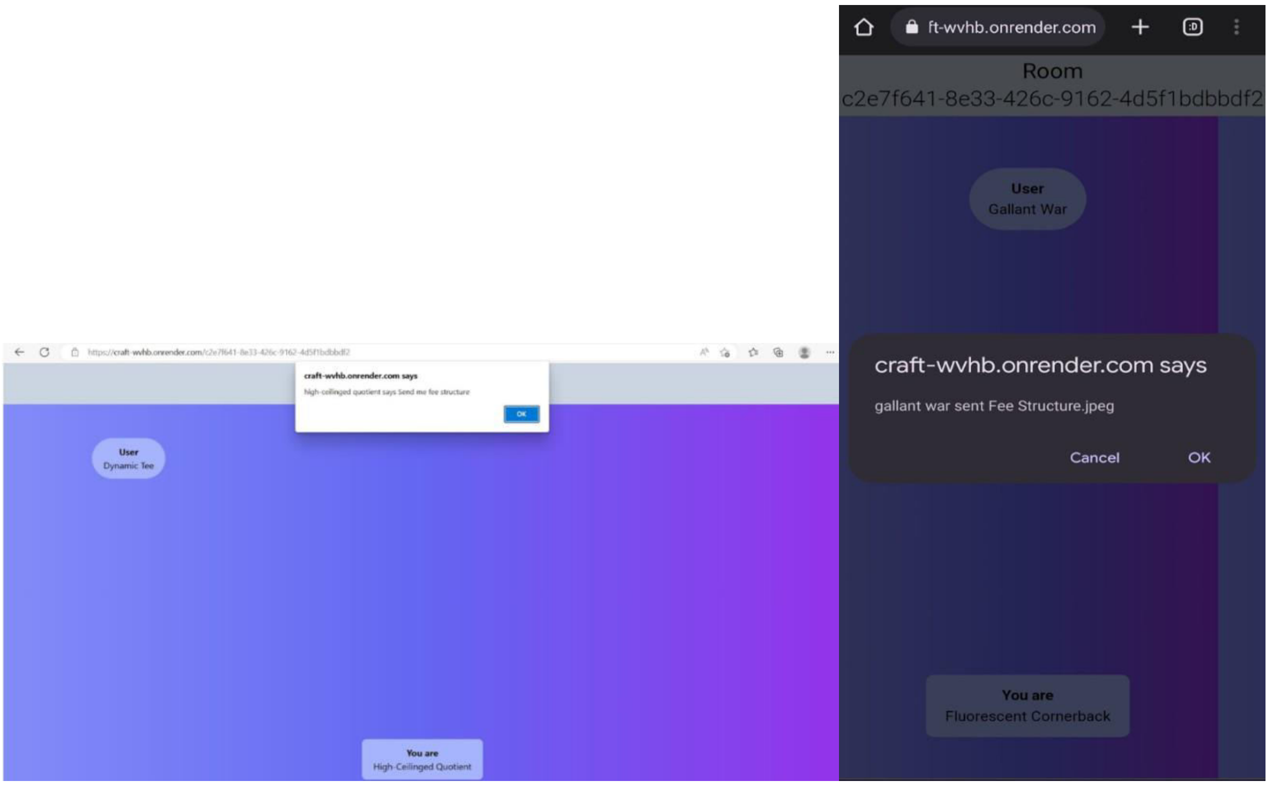
Send me timetable for exams


https://craft-wvhb.onrender.com/c2e7f641-8e33-426c-9162-4d5f1bdbbdf2
User
Dynamic Tee
OK
You are
High-Ceilinged Quotient

craft-wvhb.onrender.com says

high-ceilinged quotient says Send me fee structure

Figure 3&4: Sending & Receiving Messages



https://craft-wvhb.onrender.com/c2e7f641-8e33-426c-9162-4d5f1bdbbdf2

craft-wvhb.onrender.com says

high-ceilinged quotient says Send me fee structure

OK

User
Dynamic Tee

You are
High-Ceilinged Quotient

ft-wvhb.onrender.com

Room
c2e7f641-8e33-426c-9162-4d5f1bdbbdf2

User
Gallant War

craft-wvhb.onrender.com says

gallant war sent Fee Structure.jpeg

Cancel OK

You are
Fluorescent Cornerback

Figure 5&6 : Sending & Receiving Files

## V. Conclusion and Future Work

In conclusion, this research paper has explored the development and implementation of a peer-to-peer file sharing web app using WebRTC technology. The project demonstrates that WebRTC provides a powerful and flexible solution for real-time communication and data sharing over the internet. The app enables users to easily and securely share files directly with one another without the need for a central server or intermediary. The use of WebRTC allows for direct and efficient communication between peers, resulting in faster and more reliable file transfers. Moreover, the app's user interface is simple and intuitive, making it accessible to a wide range of users. While there are still some limitations and challenges to be addressed, such as the need for more robust security measures, this project provides a solid foundation for further exploration and development of peer-to-peer file sharing applications using WebRTC. Overall, this research highlights the potential of WebRTC technology for creating decentralized, user-controlled communication networks that prioritize privacy and security.

## References

1. Rüdiger Schollmeie,'A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications', Proceedings First International Conference on Peer-to-Peer Computing pp.0-7695-1503-7, 2001.
2. Imre Kelényi, Péter Ekler, Bertalan Forstner,' Comparison of Mobile Peer-to-peer File-sharing Clients,'2008.
3. Adel Ali Al-zebari (2014),'Peer to Peer File Sharing System,'International Journal of Scientific and Engineering Research 5(9):485.
4. Ms. M. Kiruthika, Smita Dange,'An Application Of Information Retrieval In P2p Networks Using Sockets And Metadata,'2010.
5. Yipeng Zhou, Dah Ming Chiu, John C.s. Lui (2011),'A Simple Model for Chunk-Scheduling Strategies in P2P Streaming,'IEEE/ACM Transactions on Networking.
6. Yufeng Wang, Li Wei, Athanasios Vasilakos, Qun Jin (2015),'Device-to-Device based mobile social networking in proximity (MSNP) on smartphones: Framework, challenges and prototype,'Future Generation Computer Systems.

7. Jani Hautakorpi , Gonzalo Camarillo (2015),'The Session Description Protocol (SDP) Content Attribute,' Internet Engineering Task Force (IETF).
8. Mrs. Leena I Sakri, Manjunathgouda Patil, Rajdeep Sinha, C. Bhandare, Goutham, Kunthe,'A Comparative Study and Analysis on File Sharing Applications,'International Research Journal of Engineering and Technology (IRJET), 2021.

INNO SPACE
SJIF Scientific Journal Impact Factor
Impact Factor: 8.379

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर
NISCAIR