Form – 4: Results and Conclusion

1. **Team No:** 1
2. **Project Title:** Secure File Sharing System using Blockchain
3. **Hardware Requirements**

RAM	4 GB Minimum
Processor	i3 Minimum
Hard disk	256 GB HDD Min.

4. **Software Requirements**

Technology	Python 3.6
Operating System	Windows Family
IDE	VS Code
Technology	Python, Django
Database Server	MySQL
Front Design Technology	HTML, CSS, JS

5. **Formulas Used:**

Encryption:

a. Key Generation:

- Choose two large prime numbers, p **and** q.
- Compute their product, $n = p \times q$, which will be the modulus for both the public and private keys.
- Compute Euler's totient function, $\phi(n) = (p-1) \times (q-1)$, which is used to ensure the security of RSA.
- Choose an integer e such that $1 < e < \phi(n)$ and e is coprime with $\phi(n)$. This e will be the public exponent.
- Calculate the modular multiplicative inverse of e modulo $\phi(n)$, denoted as d. This d will be the private exponent.

b. Public and Private Keys:

- The public key consists of the modulus ***n*** and the public exponent ***e***.
- The private key consists of the modulus ***n*** and the private exponent ***d***.

c. Encryption:

- To encrypt a message ***M***, the sender uses the recipient's public key **(*n*, *e*)**.
- The sender computes $C \equiv M^e \bmod n$, where ***C*** is the ciphertext.

Decryption:

a. Ciphertext Reception:

- The recipient receives the ciphertext ***C*** sent by the sender.

b. Private Key Extraction:

- The recipient possesses their private key **(*n*, *d*)**, which consists of the modulus ***n*** and the private exponent ***d***.

c. Decryption:

- To decrypt the ciphertext ***C***, the recipient computes $M \equiv C^d \bmod n$, where:
- ***M*** is the original plaintext message.
- ***d*** is the private exponent.
- ***n*** is the modulus.

d. Message Recovery:

- Once the recipient has calculated ***M***, they obtain the original plaintext message sent by the sender.

6. Smart Contract:

Pseudo Code:

```
pragma solidity ^0.8.0;
contract FileSharingContract {

    struct File {
        address owner;
        string files;
        string fileName;
        uint timestamp;
    }
```

```
    mapping(string => File) files;

    event FileUploaded(address indexed owner, string fileHash, string
fileName);

    function uploadFile(string memory _fileHash, string memory
_fileName) public {
        require(bytes(_fileHash).length > 0, "Invalid file hash");
        require(bytes(_fileName).length > 0, "Invalid file name");

        require(files[_fileHash].owner == address(0), "File already exists");

        files[_fileHash] = File(msg.sender, _fileHash, _fileName,
block.timestamp);

        emit FileUploaded(msg.sender, _fileHash, _fileName);
    }

    function getFile(string memory _fileHash) public view returns
(address, string memory, string memory, uint) {
        require(bytes(_fileHash).length > 0, "Invalid file hash");
        require(files[_fileHash].owner != address(0), "File not found");

        File memory file = files[_fileHash];
        return (file. owner, file.fileHash, file.fileName, file. timestamp);
    }
}
```

7. **Applications:**

a. Enterprise Document Management: Organizations can use the system to securely share sensitive documents among departments, teams, or external partners. This ensures that confidential information remains protected and only accessible to authorized individuals, enhancing data privacy and compliance with regulations such as GDPR.

b. Academic Collaboration: Academic institutions can utilize the platform for collaborative research projects where researchers from different institutions need to share data and documents securely. This promotes transparency, prevents data tampering, and facilitates seamless collaboration among researchers.

c. Legal Document Sharing: Law firms and legal departments can leverage the system to securely share legal documents, contracts, and case files with clients, other firms, or relevant parties. By using blockchain technology, they can ensure the authenticity and integrity of legal documents, reducing the risk of disputes or fraud.

d. Healthcare Data Exchange: Healthcare organizations can employ the system to exchange electronic medical records (EMRs) and patient information securely between hospitals, clinics, and healthcare providers. Blockchain ensures data integrity, patient privacy, and compliance with HIPAA regulations, enabling seamless sharing of medical data while maintaining confidentiality.

e. Supply Chain Management: Companies involved in supply chain management can adopt the platform to share sensitive information such as product specifications, certifications, and transaction records with suppliers, manufacturers, and distributors. By leveraging blockchain for secure file sharing, they can enhance transparency, traceability, and trust throughout the supply chain ecosystem, reducing fraud and improving efficiency.

8. **Results:**

a. Security Enhancement: Your project has successfully enhanced security in file sharing by implementing blockchain technology. Through encryption techniques and decentralized storage, files are securely transmitted and stored, reducing the risk of unauthorized access or data breaches.

b. Transparency and Integrity: By utilizing blockchain for maintaining transaction records and file metadata, your project ensures transparency and integrity in file-sharing operations. The immutable nature of blockchain prevents tampering with data, providing a reliable audit trail for tracking file activities.

c. Efficient File Operations: Performance testing indicates that your system facilitates efficient file operations, including uploading and downloading files. Users experience minimal latency and swift transaction processing, enhancing overall user experience and productivity.

d. Scalability: Your project demonstrates scalability in handling increasing user and file loads while maintaining optimal performance levels. This scalability ensures that the system can accommodate growing demands and user populations without compromising efficiency.

e. Reliability and Availability: Through rigorous testing and fault tolerance mechanisms, your project exhibits high reliability and availability. The system's uptime remains consistent, and it demonstrates resilience to failures, ensuring uninterrupted access to files and services.

9. Parameters and Comparison Table:

Parameter	Previous Method	Proposed Method
Integrated Frontend	The integration of the frontend for the previous methods for secure file sharing often relied on centralized systems, where files were stored and managed on a central server. These systems typically employed access control lists (ACLs) or role-based access control (RBAC) mechanisms to regulate user access to files. However, such centralized approaches faced challenges related to single points of failure, scalability issues, and susceptibility to security breaches.	The proposed method for the front end involves developing a user-friendly interface that allows users to interact seamlessly with the blockchain-based file-sharing system. This front end will include features such as file upload and download functionality, user authentication and access control mechanisms, transaction monitoring, and a dashboard for managing shared files. The design will prioritize simplicity, intuitiveness, and responsiveness to ensure an optimal user experience, facilitating efficient and secure file sharing within the consortium of organizations.
Encryption Techniques	The encryption techniques of previous methods were commonly used to protect file contents during transmission and storage, but they did not address the need for transparent and tamper-proof transaction records. As a result, there was a growing demand for	The current method utilizes blockchain technology for encryption, leveraging its decentralized and immutable nature to enhance security. By storing file metadata and access control information on the blockchain, alongside utilizing smart contracts

	decentralized and blockchain-based solutions to address these shortcomings and provide enhanced security, transparency, and reliability in file sharing.	for enforcing access policies, the system ensures transparent and tamper-resistant file sharing.
Blockchain Technology	In previously proposed blockchain-based approaches, it relies solely on existing blockchain frameworks like Ethereum or Hyperledger Fabric.	The system incorporates smart contracts for access control and incentivization, ensuring transparency and authenticity in file-sharing processes. This project addresses the unique requirements of secure inter-organizational file-sharing, offering a streamlined and efficient platform tailored to the needs of consortiums.

10. Conclusion:

The project concludes with the successful implementation of a secure and efficient file-sharing system leveraging blockchain technology. Through encryption, decentralized storage, and smart contracts, the system ensures the confidentiality, integrity, and availability of shared files. The system's scalability, reliability, and transparency make it suitable for diverse applications across various industries. Overall, the project sets a new standard for collaborative and secure file sharing, addressing the limitations of traditional centralized systems and paving the way for enhanced data management and exchange practices.

Signature of Supervisor: