Computer Networks

NetStore: A Distributed File Storage System using TCP Sockets and Replication

Class: TY
Group Number:

Guide:

Date:

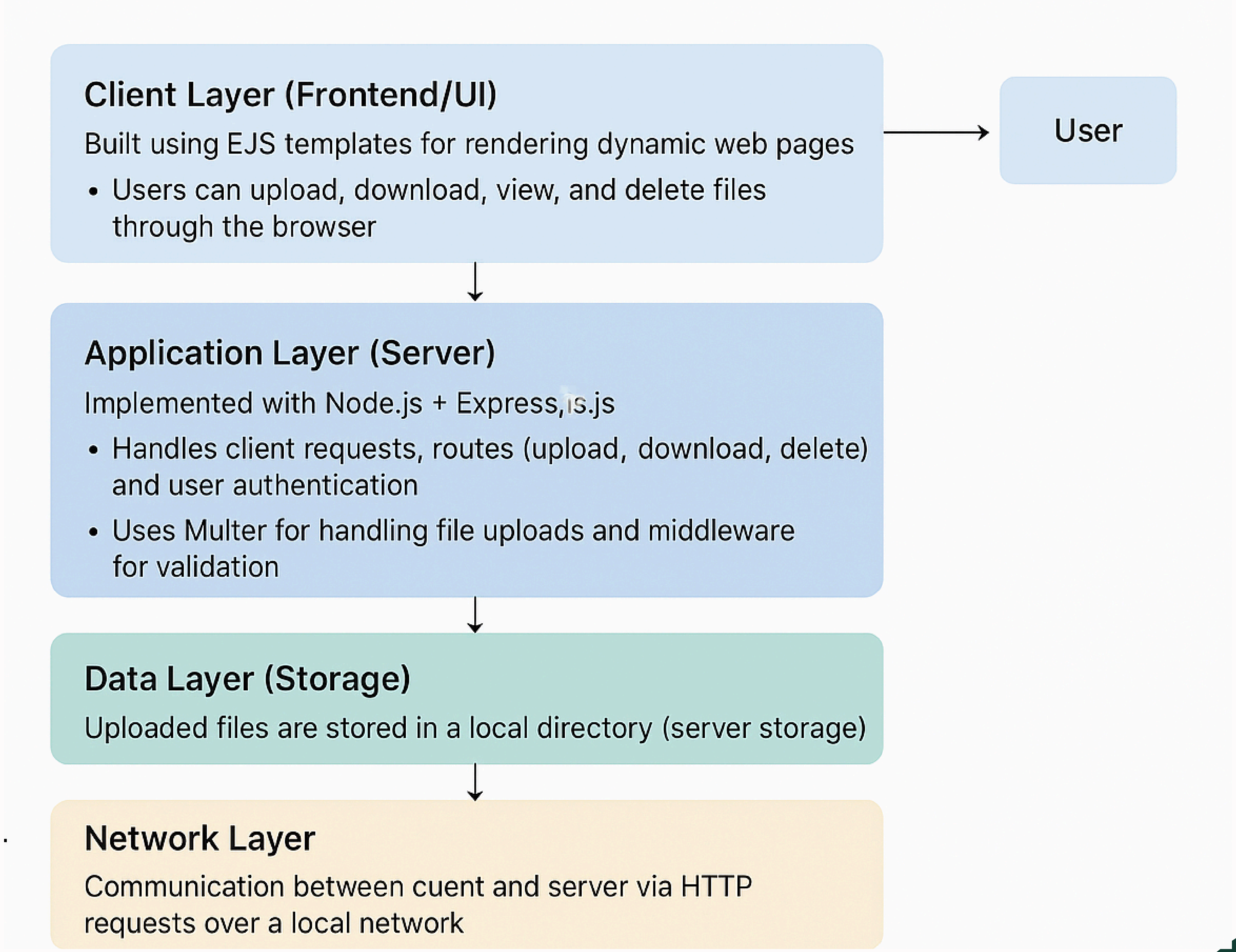
Introduction

- NetStore is a distributed file storage system ensuring data reliability and backup.
- It uses a main server with two backup servers for automatic file replication.
- Users can upload, view, download, and delete files via a Node.js web interface.
- The backend uses Python socket programming for real-time data transfer and replication.
- Includes user authentication for secure file management.

Objectives

- To develop a reliable distributed file storage system.
- To ensure data redundancy and fault tolerance using file replication.
- To provide an easy-to-use web interface for managing files.
- To enable secure access through user authentication.
- To support core file operations — upload, view, download, and delete.

Architecture



Client Layer (Frontend/UI)

Built using EJS templates for rendering dynamic web pages

- Users can upload, download, view, and delete files through the browser

User

Application Layer (Server)

Implemented with Node.js + Express,is.js

- Handles client requests, routes (upload, download, delete) and user authentication
- Uses Multer for handling file uploads and middleware for validation

Data Layer (Storage)

Uploaded files are stored in a local directory (server storage)

Network Layer

Communication between cuent and server via HTTP requests over a local network

Tech Stack

- Frontend: HTML, CSS, EJS (Embedded JavaScript Templates)
- Backend (UI): Node.js with Express.js
- Core Server: Python (Socket Programming, Multithreading)
- Database: SQLite (for Authentication)
- File Handling: Multer (for uploads) & TCP Sockets (for file transfer)

Limitations

- Limited to local network communication (no cloud integration).
- Scalability may decrease with increasing file size or number of users.
- Basic authentication — no role-based access or encryption.

Future Enhancements

- Extend storage and replication to cloud servers (AWS, Azure, GCP) for global accessibility.
- Implement load balancing and distributed databases to handle larger files and more concurrent users.
- Add role-based authentication, user permissions, and encryption for secure file transfer and access.

Conclusion

- NetStore demonstrates the practical use of TCP socket programming and client–server architecture.
- Provides reliable file storage through replication across multiple servers.
- Enhances understanding of fault tolerance, network communication, and data consistency.
- A strong foundation for building scalable and secure distributed storage systems in the future.

References

- https://www.geeksforgeeks.org/socket-programming-in-node-js/
- https://developer.mozilla.org/en-US/
- https://www.npmjs.com/
- https://www.rfc-editor.org/rfc/rfc2616

Thank You