Contents

Acknowledgement

We would like to express our deepest gratitude and sincere appreciation to all individuals who contributed directly or indirectly to the successful completion of our final year project, the “Local Area Network (LAN) Based Chat and File Sharing System.”

First and foremost, we would like to extend our sincere thanks to our project supervisors, Sr Maran Gam Awng and Sr Labya Naw Naw, for their invaluable guidance, constant encouragement, and insightful feedback throughout this project. Their expertise and patience were instrumental in helping us navigate challenges and refine our work. Without their unwavering support, this project would not have reached its full potential.

We are also immensely grateful to all the teachers and staff of Mai Ja Yang College for providing us with the necessary resources and a conducive learning environment. The knowledge and skills imparted during our coursework formed the essential foundation upon which this project was built.

Our sincere thanks go to our friends and classmates for their moral support, stimulating discussions, and for willingly testing our application. Their constructive criticism and valuable suggestions were crucial in improving the system’s functionality and user experience.

Lasty, and most importantly, we wish to express our heartfelt gratitude to our families. Their endless love, unwavering belier in our abilities, and immense patience during this demanding period were our greatest source of strength and motivation. This achievement is as much theirs as it is ours.

Thank you all…

Abstract

- This project is designed to provide chat and file sharing within a Local Area Network (LAN).
- It can operate without an internet connection, ensuring independent local communication.
- The system is developed using socket API, supporting both client-server and peer-to-peer communication.
- It enables real-time messaging and fast file transfer with minimal delay.
- A user-friendly interface is implemented to make the system easy to use.

CHAPTER 1

Introduction

1.1 Project Overview

Local Area Network (LAN) Based Chat and File Sharing System is designed to enable real-time communication and file sharing among computers connected within a LAN. The system allows data exchange between users without requiring an internet connection, making it efficient and secure for local communication.

The system is to facilitate easy and fast communication and file transfer among team members, departments, office and within a classroom environment. The system enables instant messaging between connected users within the LAN and sending and receiving files (documents, images, etc.) between users or group. Secure login with Username and Password to ensure data privacy. Provides sound or visual alerts for incoming messages and displays online/offline status of user.

This system uses Python language in the backend and HTML/CSS, JavaScript, Json in the frontend. TCP/IP socket programming is built on the Linux Platform.

Develop a chat system that work efficiently over a LAN without the need for internet access and to make file sharing among team members easy and secure. To design a user-friendly interface and reliable communication platform. Lacal Area Network (LAN) Based Chat and File Sharing System provide a standalone LAN-based system capable of real-time messaging and high-speed file transfer among connected computers.

1.2 Objective of the project's

This project has several main goals. Its first objective is to provide fast and efficient real-time communication for users on the same local network (LAN). It also aims to make sharing files between these users a simple and quick process. Another important function is to allow for the submission of activities, such as school assignments. The system is designed to support direct one-to-one, or peer-to-peer, and group conversations. Finally, all of these features will be accessible through an easy-to-use and user-friendly interface, ensuring a smooth experience for everyone.

1.3 Project Scope and Limitations

1.3.1 Project Scope

"Project Scope" refers to what is included and what the system is capable of doing.

1. User Management

Registration: Users can register an account using a Username and Password.

User List: Ability to view a list of all users currently on the network.

User Status: Ability to see the status of users (e.g., Online, Offline).

2. Real-time Messaging

Private Chat: Users can send private messages to each other.

Group / Broadcast Chat: Users can connect in groups and send messages to all users simultaneously.

Chat History: Users can view their message history.

3. File Sharing

Users can send files, photos, and videos to each other. Supports direct file transfer. File size limits are defined.

4. Administration & Network

Local Network Operation: The system works only within the same local area network (via Wi-Fi or Ethernet).

Centralized Control (Client-Server Model): A central server manages all information and operations.

1.3.2 Project Limitations

"Limitation" refers to the weaknesses of this system or the things it cannot do.

Performance

Limited Users: Depending on the server's performance capacity, the system may become slow if the number of users exceeds 20-30-50. It may not be possible to support over 100 users. Currently, the video call feature is not yet available.

Large File Issues: There may be restrictions in place that prevent sending very large files, such as those several GB in size or videos.

Memory Usage: Storing users' files and chat history requires significant use of the server's memory (RAM).

CHAPTER 2

Background Study

2.1 What is Local Arae Network (LAN)?

A Local Area Network (LAN) is a type of network created to connect computers, printers, phones, and other computer devices located close to each other in a single location (for example, one home, one office, one school) so that they can communicate and be used together.

For example

· If you have Wi-Fi at home, all the phones, computers, and Smart TVs connected to that Wi-Fi are within your home's LAN.

· A computer lab at school with about 50 connected computers, or classrooms with Wi-Fi, are also LANs.

2.1.1 Where are LANs used?

You can find LANs almost everywhere.

1. Homes (Home Networks) - For family members to use the internet, share files, and play games together.

2. Schools - In computer labs, libraries, and teachers' offices.

3. Offices - For employees to communicate with each other, share data, and use a central printer.

4. Hospitals - For patient lookup systems and sharing medical records.

5. Public places like coffee shops - For customers to use the internet.

2.1.2 What is a LAN for? Why is it important?

The main purpose of a LAN is to share resources and make communication faster and easier.

1. Shared Internet Access - One internet line can be used by all computers and phones in an entire house.

2. File and Data Sharing - You can wirelessly send schoolwork from one computer to another. In an office, a document can be set up for all team members to view.

3. Hardware Sharing - You don't need 20 printers for 20 computers. All computers can connect to and use a single printer.

4. Easy Communication - Using chat software (e.g., LAN Messenger) on a LAN allows students or office team members to send messages to each other instantly.

5. Multiplayer Gaming - You can play games together with friends in a computer lab.

2.1.3 How does it benefit students?

For a student, a LAN greatly supports your education and daily life.

1. Research & Learning - Using the school's Wi-Fi, you can easily access the internet to search for information and access online libraries and e-books.

2. Collaboration - Using online platforms like Google Docs or Microsoft Teams via the LAN, you can work on group essays, prepare presentations, and easily send files to other students.

3. Resource Efficiency - To install software in a school computer lab, it doesn't need to be installed on every single computer. It can be distributed to all from one central server.

4. Knowledge Expansion - Understanding LANs provides a good foundation for students interested in technology fields like networking and IT management.

5. Easy Management - Teachers can centrally manage students' computers through the LAN to deliver lessons and assign exercises.

2.2 System Requirements

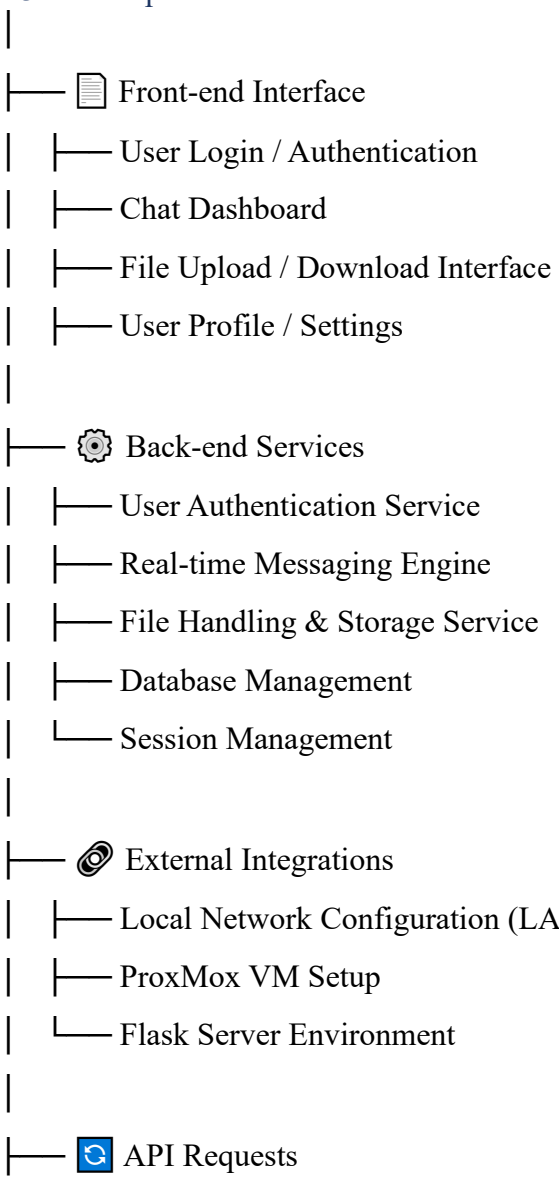
Hardware

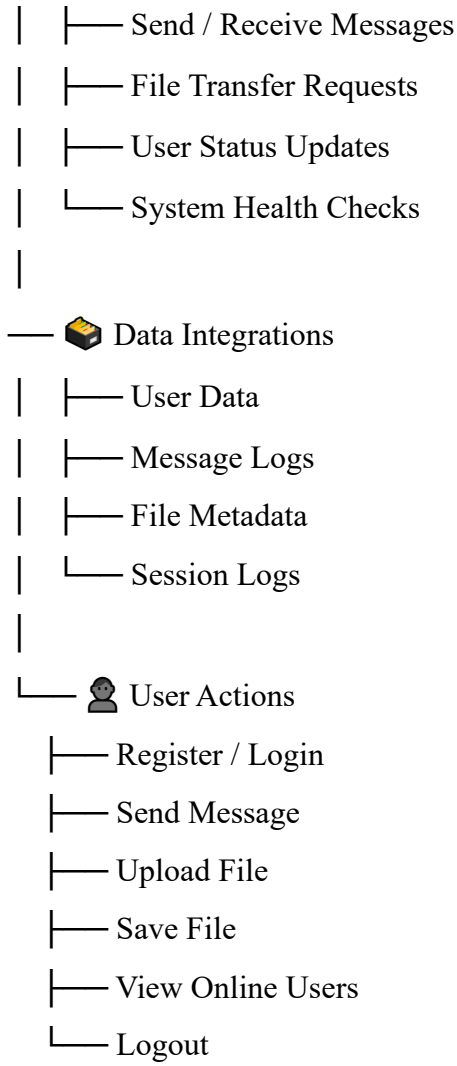
- Proxmox Server: Run multiple VM containers with sufficient CPU, Ram and Storage.
- Local network and router: Support LAN distribution with static IP configuration for each service.
- Ethernet cable: support for network or Wifi, connect to router.

Software

- Proxmox VE: Updated to the latest version.
- Linux VM Templates: Use Debian for each container (Flask, File Browser)

2.3 Site Map





2.3.1 Front-end Interface Layer

The front-end serves as the primary point of interaction for the end-user. It is a dynamic web application that renders the user interface and communicates with the back-end services via a defined API. Its key modules include:

- **User Login / Authentication Interface:** A secure portal that collects user credentials and manages the login/logout life cycle.
- **Chat Dashboard:** The central hub for real-time communication, displaying active conversations, contact lists, and message histories.
- **File Upload / Download Interface:** Provides a drag-and-drop or browser-based mechanism for users to transfer files to and from the system.
- **User Profile / Settings Module:** Allows users to view and modify their personal information and application preferences.

2.3.2 Back-end Services Layer

This layer constitutes the core application logic, built upon the Django web framework. It is composed of several discrete services:

- **User Authentication Service:** Validates user credentials, manages password hashing, and issues session tokens to maintain user state.
- **Real-time Messaging Engine:** Leverages Web Sockets to facilitate instantaneous, bi-directional communication for message delivery and user status updates.
- **File Handling & Storage Service:** Manages the secure reception, storage, retrieval, and access control of user-uploaded files on the server's file-system.
- **Database Management Service:** Handles all Create, Read, Update, and Delete (CRUD) operations for persistent data, abstracting direct database interactions.
- **Session Management Service:** Tracks active user sessions, manages timeouts, and ensures secure access to protected resources.

2.3.3 External Integrations and Deployment Environment

The system's operational context is defined by its integration with specific external platforms and network configurations:

- **Local Network Configuration (LAN):** The system is designed to operate within a LAN, leveraging its high-bandwidth, low-latency characteristics for optimal performance and enhanced security through network isolation.
- **Proxmox LXC Setup:** The server component is containerized within a Proxmox Linux Container (LXC), providing a lightweight, isolated, and easily manageable runtime environment.

2.3.4 API Communication Layer

The interaction between the front-end and back-end is standardized through a Restful API and Web Socket connections. The primary API endpoints and actions include:

- **Message Endpoints:** For sending and receiving text messages in real-time.
- **File Transfer Endpoints:** For initiating uploads and downloads, handling multipart/form-data requests.
- **User Status Endpoints:** For updating and fetching the online/offline status of users.
- **System Health Check Endpoint:** A monitoring endpoint used to verify the operational status of the server and its services.

2.3.5 Data Integrations and Persistence

The system interacts with a relational database to persistently store all operational data. The key data entities are:

- **User Data:** Stores user profiles, hashed passwords, and preferences.
- **Message Logs:** Archives all sent and received messages with timestamps and sender/receiver identifiers for history and auditing.
- **File Metadata:** Contains information about stored files, such as filename, size, uploader, upload timestamp, and physical storage path.
- **Session Logs:** Records user login and logout times, IP addresses, and session tokens for security and analytics.

2.3.6 User Interaction Flow

The system is designed around a set of core user actions that define the primary use cases. These actions trigger the aforementioned components in a coordinated sequence:

1. **Register / Login:** A user provides credentials, which are verified by the Authentication Service, leading to session creation.
2. **Send Message:** A message composed in the Chat Dashboard is sent via the API to the Real-time Messaging Engine, which broadcasts it to the intended recipient(s).
3. **Upload File:** A file selected through the interface is transmitted to the File Handling Service, which stores it and records its metadata in the database.
4. **Save File:** A user request to download a file is routed through the File Handling Service, which retrieves the file from storage and initiates the download.

CHAPTER 3
System Methodology

3.1 ProxMox VM, Flask and Networking Setup

LXC is an operating-system-level virtualization environment for running multiple, isolated Linux system on a single Linux control host. VM works as a userspace interface for the Linux kernel containment feature. Users can easily create and manage system or application containers with a powerful API and simple tools.

To create a stable, isolated, and easily manageable development and deployment environment, a virtualized infrastructure was implemented using Proxmox Virtual Environment (VE). The core application was then developed using the Django web framework and containerized for consistency.ပြန်ဝရုံးရန်

3.1.1 Proxmox Virtual Environment (VE) Setup

Proxmox VE was chosen as the primary platform for virtualization. It was installed on a dedicated server to host all the project's virtual machines (VMs) and Linux Containers (LXC). Proxmox provides a powerful web-based dashboard for managing virtualized resources, which simplified tasks such as creating, snapshotting, and managing our containers.

- **Initial Configuration:** The Proxmox host server was configured with a static IP address on the college LAN to ensure consistent accessibility for all team members. Storage was set up and named appropriately (e.g., local-lvm).
- **Cluster (Optional):** A single-node cluster was established, as the project did not require high availability.

3.1.2 LXC Container Deployment

Instead of full virtual machines, lightweight Linux Containers (LXC) were used to host the application. This approach offers superior performance and lower overhead while maintaining strong isolation.

- **Container Template:** A Debian 12 LTS template was downloaded from the Proxmox repository and used as the base for creating a new LXC container.
- **Container Creation:** A new unprivileged LXC container was created with the following resource allocation:
    - **CPU Cores:** 2

- **Memory (RAM):** 2048 MB
- **Disk Space:** 20 GB

- **Network Configuration:** The container was connected to the vmbr0 bridge, which is the default virtual bridge in Proxmox connected to the physical network. A static IP address within the college LAN's DHCP range (e.g., 192.168.1.50) was assigned to the container to ensure it was always reachable at the same address by clients on the network.

3.1.3 Application Environment Setup inside LXC

After starting the container, the environment for the Django application was prepared.

1. **System Update & Package Installation:** The container's package list was updated, and essential packages like python3-pip, python3-venv, and git were installed.

```
##bash
        >apt update && apt upgrade -y
        >apt install python3-pip python3-venv git -y
```

3.2 Phyton

Python is a programming language that is widely used in web applications, software development, data science, and machine learning (ML). Developers use Python because it is efficient and easy to learn and can run on many different platforms.

3.3 JavaScript

JavaScript is a scripting language that enables you to create dynamically updating content, control multimedia, animate images, and pretty much everything else. i.e. anything that moves, refreshes, or otherwise changes on your screen without requiring you to manually reload a web page. Features like: animated graphics. photo slideshows.

3.4 HTML/CSS

HTML is a markup language used to create static web pages and web applications. CSS is a style sheet language responsible for the presentation of documents written in a markup language. Consists of tags surrounding content. For Example: *Welcome to Simplilearn*

3.5 JSON

JavaScript Object Notation (JSON) is a standard text-based format for representing structured data based on JavaScript object syntax. It is commonly used for transmitting data in web applications (e.g., sending some data from the server to the client, so it can be displayed on a web page, or vice versa).

CHAPTER 4

System Design

4.1 Use Case Diagram

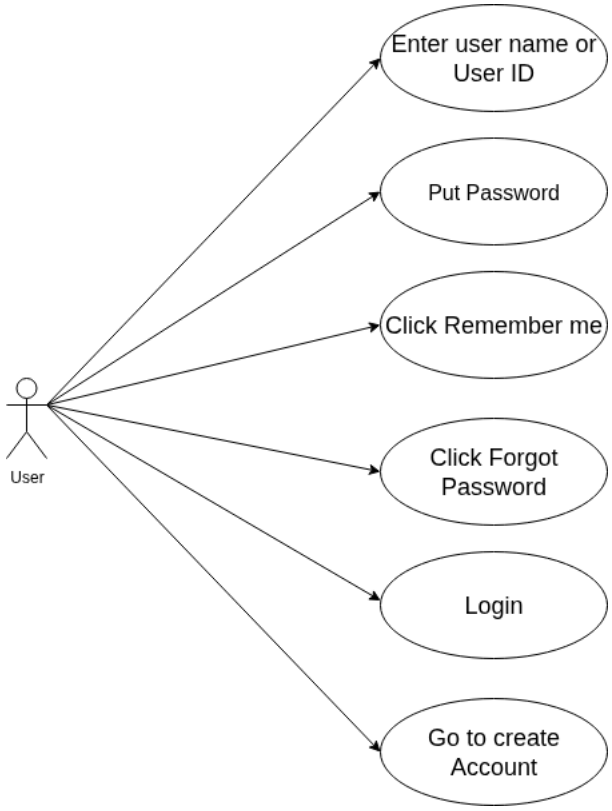


Figure1.User Login Form

CHAPTER 5
Cost Estimation