**BHARATIYA VIDYA BHAVAN’S**

**SARDAR PATEL INSTITUTE OF TECHNOLOGY**

**(Autonomous Institute Affiliated to University of Mumbai)**

MUNSHI NAGAR, ANDHERI (WEST), MUMBAI – 400058

**Department of Computer Engineering 2023-24**

**Distributed Computing Lab**

****

**LinkUP : A real time collaboration system**

**Instructor : Prof. Mayuri More**

Nis*hita Panchal | 2022300073*

*Manjiri Chavande | 2023301003*

*Chetan Patil | 2023301012*

*Akhil Pillai | 202230008*2

***TE Comps B | Batch A***

**LinkUp**

### 

### **Project Scope**

#### **1.1. Project Overview**

The project involves developing LinkUp, a real-time messaging and collaboration tool for teams. The application will offer features like instant messaging, video and voice calls, channel-based communication, and workspace management. The platform aims to facilitate seamless communication and collaboration among team members within an organization, mimicking the core functionalities of Slack.

#### **1.2. Objectives**

* **User Authentication**: Implement secure sign-up and login functionality using Supabase with options for Google, GitHub, and Email authentication.
* **Real-Time Communication**: Enable real-time text-based chat, voice calls, and video calls using technologies like socket.IO, Websockets, and WebRTC.
* **Channel and Workspace Management**: Allow users to create and manage channels and workspaces, supporting both public and private channels.
* **Role Management**: Differentiate user roles between Admin and User, with Admins having the ability to manage channels, workspaces, and user permissions.
* **Data Persistence**: Store and manage user data, chat messages, and other related information in Supabase, utilizing its capabilities for RPC, storage, SQL, and role-level security.
* **Scalability and Security**: Ensure that the platform is scalable to handle multiple users and channels, with robust security features to protect user data.

#### **1.3. Deliverables**

* **Frontend Interface**: A responsive web application with a user-friendly interface for accessing the platform's features.
* **Authentication System**: Integration with Supabase for secure user authentication.
* **Chat System**: Real-time chat service with support for individual and group conversations within channels.
* **Voice and Video Calls**: Implementation of voice and video call features using WebRTC.
* **Channel and Workspace Management**: Functionality for creating and managing channels and workspaces, with options for public and private settings.
* **Admin Dashboard**: An interface for Admins to manage users, channels, and workspaces.

### **Problem Statement**

Teams often struggle with fragmented communication and high costs associated with existing tools like Slack. Small to medium-sized organizations need a cost-effective, customizable, and self-hosted platform that integrates real-time messaging, voice and video calls, and workspace management. Our project aims to develop LinkUp to meet these needs, providing an efficient, scalable, and secure solution for seamless team collaboration.

### **2.1 Objectives**

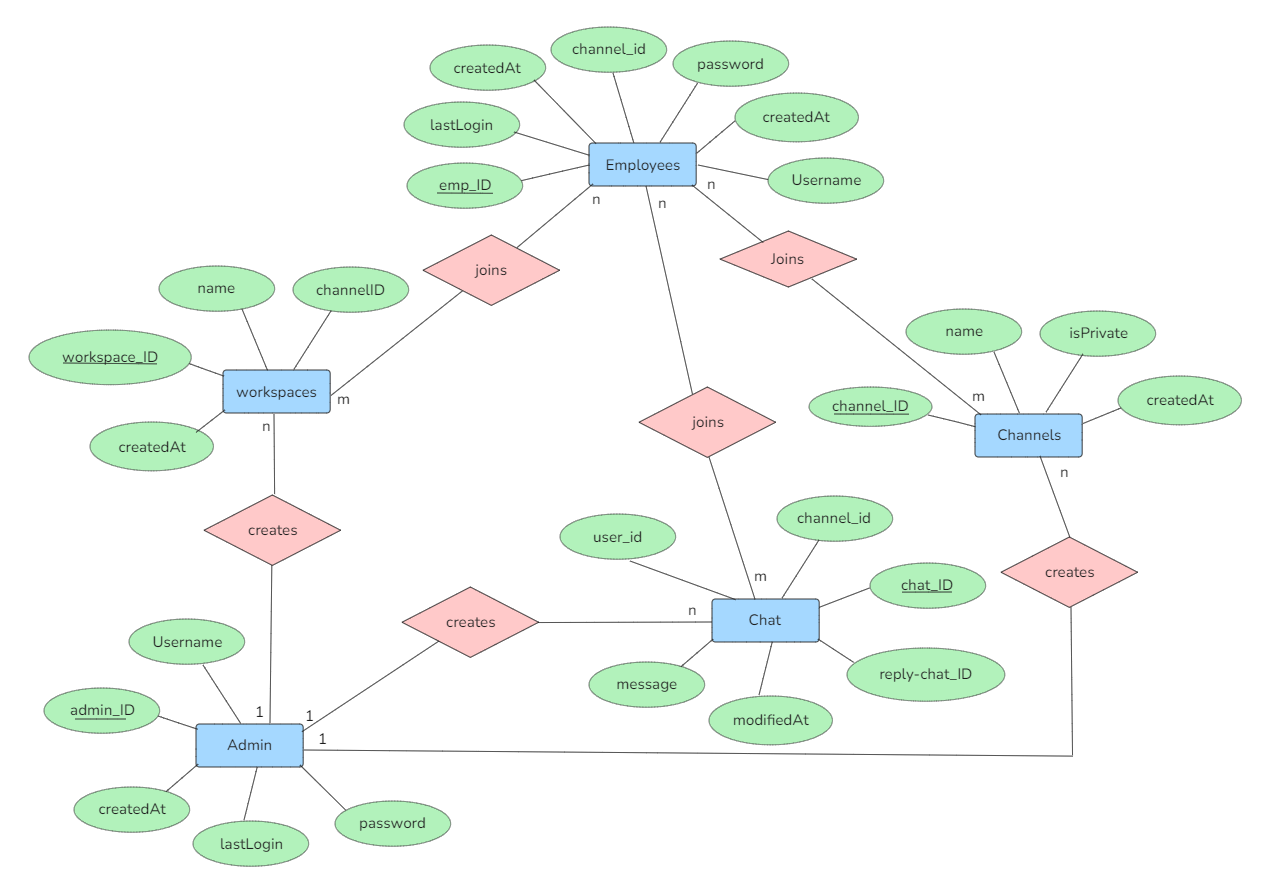
1. **User Authentication**:
   * Implement secure sign-up and login options using Google, GitHub, and Email via Supabase.
2. **Real-Time Communication**:
   * Enable real-time text-based messaging, voice calls, and video calls using socket.IO, Websockets, and WebRTC.
3. **Channel and Workspace Management**:
   * Allow users to create, manage, and organize channels and workspaces, with support for both public and private settings.
4. **Role-Based Access Control**:
   * Differentiate between Admin and User roles, where Admins can manage users, channels, and workspaces.
5. **Data Persistence and Security**:
   * Store and manage user data, chat messages, and related information securely using Supabase, ensuring role-level security.
6. **Scalability**:
   * Design the system to handle multiple users, channels, and workspaces efficiently, ensuring smooth performance as the user base grows.
7. **User-Friendly Interface**:
   * Develop a responsive and intuitive interface for seamless user interaction across all devices.
8. **Admin Dashboard**:
   * Provide Admins with tools to monitor and manage the platform, including user management, channel/workspace settings, and data analytics.
9. **Customizability**:
   * Ensure the system can be tailored to fit the unique needs of different organizations, allowing for future enhancements and integrations.

### **3. Functional Requirements**

1. **User Authentication**:
   * Users must be able to sign up and log in using Google, GitHub, or Email.
   * Password reset functionality should be available.
2. **User Roles and Permissions**:
   * Admins must be able to create, modify, and delete channels and workspaces.
   * Admins should manage user permissions within channels and workspaces.
   * Users should have the ability to join and leave channels and workspaces.
3. **Real-Time Messaging**:
   * Users must be able to send and receive real-time text messages within channels.
   * The system should support direct messaging between users.
   * The chat system must support multimedia messages (images, files).
4. **Voice and Video Calls**:
   * Users should be able to initiate and join voice calls within channels or direct messages.
   * Users must be able to initiate and join video calls.
5. **Channel and Workspace Management**:
   * Users must be able to create public and private channels.
   * Users should be able to create and manage workspaces.
   * The system should allow for inviting users to specific channels and workspaces.
6. **Notifications**:
   * Users should receive real-time notifications for new messages, mentions, and calls.
   * Admins and users must be able to customize notification preferences.
7. **Data Persistence and Storage**:
   * All messages, user data, and files must be stored securely in the database.
   * Users should have access to their chat history and media files.
8. **Search Functionality**:
   * Users must be able to search for messages, users, and files within channels and workspaces.
9. **Security Features**:
   * The system must enforce role-based access control for different user roles.
   * All data should be encrypted both in transit and at rest.
   * The system must support two-factor authentication (2FA) for enhanced security.
10. **Admin Dashboard**:
    * Admins should have a dashboard to manage users, monitor channel activity, and view analytics.
    * The dashboard should include tools for managing system settings and user roles.
11. **Scalability**:
    * The system must handle a growing number of users, channels, and messages without performance degradation.
12. **Integration with Third-Party Services**:
    * The system should allow integration with third-party services like calendars, task management tools, or external file storage systems (optional but recommended for future versions).

### **4 . Non-Functional Requirements**

1. **Performance**:
   * The system should respond to user actions within 2 seconds for typical operations (e.g., sending a message, switching channels).
   * The system must handle up to 10,000 concurrent users without noticeable performance degradation.
2. **Scalability**:
   * The architecture should support horizontal scaling to accommodate an increasing number of users, channels, and workspaces.
   * The system must be capable of scaling to support large organizations with thousands of users.
3. **Security**:
   * All user data must be encrypted in transit (using HTTPS) and at rest.
   * The system should implement role-based access control (RBAC) to restrict unauthorized access to certain features and data.
   * Support for two-factor authentication (2FA) should be implemented for enhanced user security.
   * Regular security audits and vulnerability assessments must be conducted.
4. **Reliability**:
   * The system should have 99.9% uptime, with minimal unplanned outages.
   * It must include mechanisms for automatic recovery from failures (e.g., database backups, server redundancy).
5. **Usability**:
   * The user interface must be intuitive, with a consistent design across all features.
   * The system should be accessible, adhering to standard accessibility guidelines (e.g., WCAG 2.1).
   * It should support multiple languages for a global user base.
6. **Maintainability**:
   * The codebase must be well-documented, with clear guidelines for adding new features or making modifications.
   * The system should support continuous integration and continuous deployment (CI/CD) to allow for rapid updates and bug fixes.
7. **Compatibility**:
   * The web application must be compatible with all major browsers (Chrome, Firefox, Safari, Edge).
   * The system should be responsive and work seamlessly across different devices (desktop, tablets, smartphones).
8. **Data Consistency**:
   * The system must ensure that data is consistent across all users and channels, with no discrepancies or data loss during operations.
9. **Data Retention and Backup**:
   * The system should implement regular backups of all user data, with the ability to restore data in case of failure.
   * Data retention policies must be configurable to meet organizational needs (e.g., for compliance purposes).
10. **Logging and Monitoring**:
    * The system should include comprehensive logging for tracking user actions, system events, and errors.
    * Real-time monitoring tools should be implemented to detect and respond to issues promptly.
11. **Latency**:
    * The system should minimize latency for real-time communication features, ensuring voice and video calls have low lag (under 100ms for optimal experience).
12. **Disaster Recovery**:
    * A disaster recovery plan must be in place to ensure that the system can recover from catastrophic events with minimal data loss and downtime.

**ER diagram :**

**System Design:** 