ecosmob
bringing innovative technologies

Supporting backend development for Audio, Video(Call & Conferencing) & Chat features for Desktop/Mobile App

Date- 13-02-2024

Scope of Work for Codiis

ECOSMOB TECHNOLOGIES PVT. LTD.

www.ecosmob.com

Table of Contents:

Project Summary

Client Name:-

Gnanaprakash

Organization Name- Codiis

Project Name:-

SIP server development

What is the Goal?

The goal of this project is to create a comprehensive communication platform by developing a robust SIP server to support both smartphone and desktop applications. These applications will offer various communication features including messaging, audio calls, and video calls. Additionally, the project aims to enhance security by implementing encryption methods to safeguard communication channels.

What problem does this solve?

The project aims to solve the challenges associated with communication and collaboration by providing a secure, interoperable, and scalable platform for messaging, audio calls, and video conferencing. It empowers users to communicate effectively and efficiently, leading to improved productivity and collaboration outcomes.

When to launch this solution and why?

NA

Who is Going to use the requested solution?

Prospect and their end users

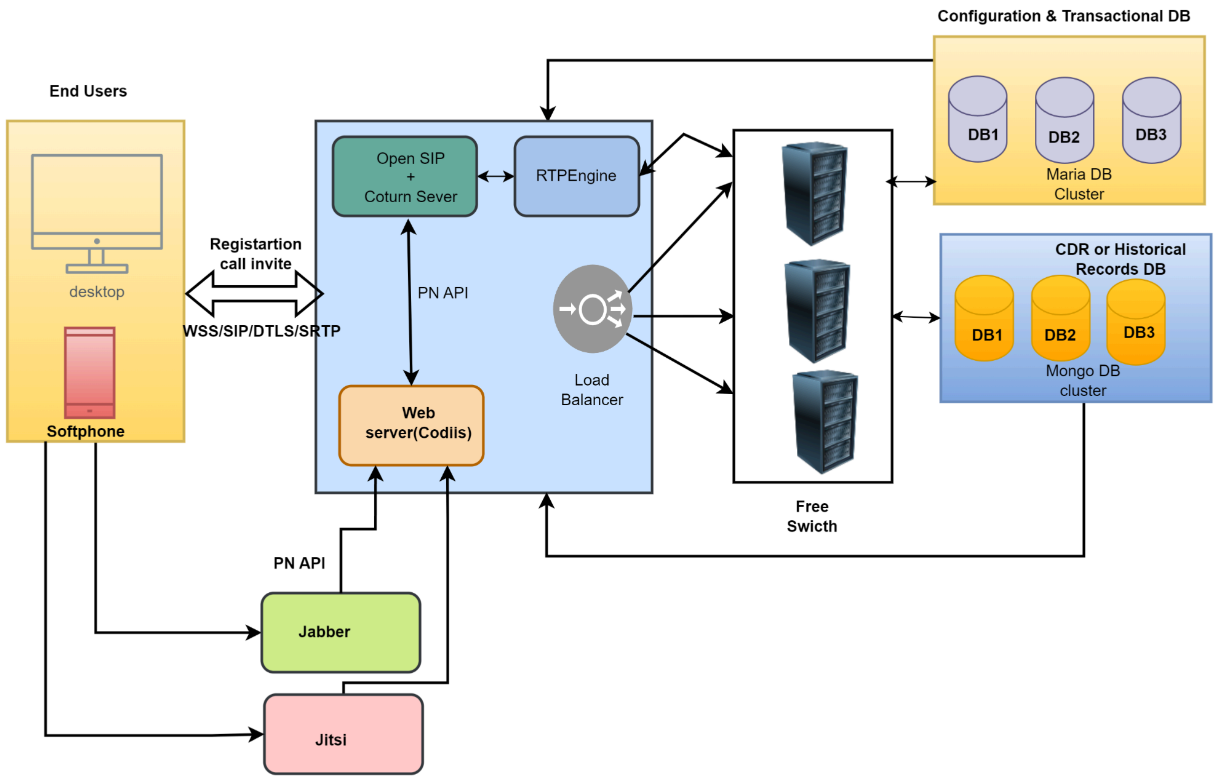
Project Overview

The project aims to develop a robust backend system supporting Softphone and desktop applications which provide messaging, audio, and video call capabilities, incorporating encryption methods for secure communication. The server will utilize SIP functionality for audio/video, XMPP for chat, and Jitsi for video conferencing.

Features & Specifications

- The goal is to create the backend support for below listed features:
    - **Audio Calls**: Support for audio calls categorized as following,
        - User to User call
        - User to PSTN Call
        - PSTN to User Call
    - **Video**: Video calling mainly categorized into **Video call & Video conferencing** for Desktop & Mobile applications.
    - **Chat Functionality**: This involves allowing users to exchange text messages in real-time. It's like sending instant messages back and forth. It allows users to make one-to-one as well as group chats.

High Level Architecture



Feature Specifications:

Coturn Server:

- Coturn is an implementation of the TURN (Traversal Using Relays around NAT) and STUN (Session Traversal Utilities for NAT) server protocols.
- **STUN Server :** STUN servers assist in establishing direct peer-to-peer connections between devices in situations where they may be behind Network Address Translation (NAT) devices or firewalls. STUN servers help by:
  - Discovering the public IP address and port of a device when it's behind a NAT.
  - Detecting the type of NAT in use, which is essential for selecting appropriate connection methods.
  - Facilitating the traversal of NAT devices by keeping NAT bindings open.
- **TURN Server:** When direct peer-to-peer connections are not possible due to restrictive network conditions or firewalls, TURN servers act as intermediaries.

Their roles include:

- Relaying media traffic between devices, ensuring communication even when direct connections are blocked.
- Serving as a fallback mechanism when STUN fails to establish direct connections.
- Encrypting the relay data to ensure privacy and security.

OpenSIP as WebRTC Gateway:

- For the WebRTC Softphone, OpenSIPs will work as a WebRTC Gateway Establishes real time communication over a secure websocket(WSS) connection.
- WebRTC WSS ←→ SIP Signaling Conversion
- Provides NAT Traversal
- Use OpenSIP+RTPEngine as WebRTC ←→ SIP Gateway

RTP Engine:

- RTP engine in a WebRTC setup is responsible for the efficient and reliable transport of real-time media handling between participants in a communication session.
- It handles packetization, codec management, error handling, and synchronization, all while working seamlessly within the WebRTC stack to deliver high-quality, secure, and real-time communication experiences.

OpenSIP as Load balancer:

- **Registration Request:**
    - Here, OpenSIP acts as a Mid Registrar - SIP Registration Forwarding to FreeSwitch Servers
    - End Softphone will send a Registration request to the OpenSIPs server, further Load balancer in OpenSIP Server will send the registration request to the specific freeswitch server according to the load balancing logic (round robin). Thereby distributing load equally among all freeswitch servers in the backend.
    - Freeswitch will process the same registration request and authorize the customer for registration.
- **Call Request:**

- Registered customers can make and receive audio/video calls using their softphone application.
- When a softphone user makes an outbound call request, it will be received at openSIP in WSS format. Further openSIP translates requests to SIP signals and load balance to freeswitch servers in the backend to handle calling features. Same way it works for inbound calls.

OpenSIP as Push Call Server-Why do we use it?

- To handle the SIP Registration and webRTC Conversation
- User and Remote SIP Server Management etc.
- Notification delivery of different events of call and instant messages to softphone application users.
- Signaling messages are exchanged between devices via the push call server to establish a direct peer-to-peer connection using WebRTC.
- Once established, audio and video data can be transmitted directly between the caller and recipient, bypassing the push call server. Either party can end the call, with the softphone app sending a message to the push call server to update the call state.
- Call Notifications,message using Push Call Server

Video conferencing features:

- **Video Conferencing**: Allows users to conduct video conferences with multiple participants, facilitating real-time communication and collaboration.
- **Audio Conferencing**: Audio conferences, enabling users to join meetings via phone or computer audio.
- **Chat**: Chat functionality within the conferencing interface, allowing participants to exchange text messages during meetings.
- **Participant Management**: Hosts can manage multiple participants, including muting/unmuting and many other features.
- **Recording:** Allows users to record video conferences for later reference or sharing with participants who couldn't attend the live session.
- **Live Streaming:** Live streaming of video conferences enabling larger audiences to view the conference in real time.
- **Security:** End-to-end encryption for video and audio communication, ensuring privacy and security during conferences.
- **Password Protection and Waiting Rooms**: Hosts can configure video conferences

to require a password for entry, adding an extra layer of security. Additionally, waiting rooms can be enabled, allowing hosts to admit participants individually, which can help manage access to the conference.

Chat:

- Chat functionality typically involves text-based messaging between app users.
- Chat will include one to one chat, Group chat, Media transfer such as Image and File transfer, Presence etc.
- The client sends chat messages to the server using a protocol like XMPP (eXtensible Messaging and Presence Protocol).
- The chat messages are transmitted over the network using a standard XMPP protocol (Jabber).

Backend Features

- **ICE/STUN/TURN Server**
- **Audio/Video call**
    - User to User to call
    - User to PSTN call
    - PSTN to User call
- **Video Conference Meeting:** Schedule and join video conferences with multiple participants.
- **One-on-One Chat:** Send and receive private messages with other users.
- **Group Chat:** Create and participate in group chats with multiple users.
- **Call History:** View call history and details of past calls.
- **Meeting History :** View meeting records and details of past meetings.
- **Presence:** View online status of other users.
- **Push Notifications:** Receive real-time alerts for incoming calls, messages, and conference updates.
- **Media Sharing:** Send and receive media files over chat
- **Media Storage:** Store shared media on storage system
- **Screen sharing:** Screen sharing during meetings
- **Breakout room**
- **Waiting or Lobby for Video meeting**
- **Media Codec Transcoding**
- **Encryption for Data at Rest**

- **TLS for Data at Transmit**

Technology Stack

Modules	Technology
Desktop Backend	Electron.JS
Mobile App	React Native
Web Server	NGINX
SIP PBX Server + Routing Engine	FreeSwitch
Push Call Server + Load Balancer + WebRTC Gateway + SIP Proxy	OpenSIP
Database	MariaDB
CDR Database	MongoDB
Video conferencing	Jitsi
Chat	Jabber(XMPP)

Notes/Assumptions:

- In this scope, we have not considered development for the frontend services of Desktop & Mobile App.
- We are considering the above technology stack for development and integration with the frontend side. Kindly review the same and share your feedback.
- Ecosmob will be responsible for the integration, development, and deployment of this solution on the prospect's server.
- The prospect will be responsible for the cost of the servers.
- The Prospect has to manage procurement of DID & SIP Trunk.
- Proposed architecture is high level design and may be subject to change.
- Ecosmob will not be responsible for any 3rd party service or API procurement and its license or subscription cost.

- Please evaluate the scope and let us know if any features should be added, removed, or modified.

Query:

- Detailed information required about features consideration for development of PBX core services?
- Do we need to develop a front end portal for core PBX’s feature management?
- Does this solution require zero touch provisioning for Desktop and Mobile App users?
- Please evaluate the scope and share your feedback for considered features.