CSC3002 Proposal of Group 52

VidSync: A C++ Based Instant Video Chat Application

**Introduction:**

For our course project, we choose to develop an instant video chat software, which we call it VidSync. The core idea of VidSync is to provide a seamless and efficient real-time communication platform that supports both text-based and video-based interactions. We chose this topic because video communication has become essential in the modern digital era, driven by remote work, online education, and social connections. Our project aims to create a lightweight yet powerful application that allows users to communicate over video and chat with minimal latency, leveraging Qt for GUI development, Redis for real-time messaging, and MySQL for data persistence. This project not only aligns with current technological trends but also provides us with an opportunity to explore a range of software engineering topics, such as network communication, multimedia handling, and database integration.

**Related work:**

Our ideas draw inspiration from popular video communication applications like Zoom, Google Meet, and Tencent Meeting. These platforms are designed to provide real-time communication with low latency, supporting video calls and text chat. Although these applications are robust and feature-rich, they are often complex and resource-intensive. In contrast, VidSync will focus on simplicity, performance, and seamless integration using C++ as the core language and Qt for the user interface. And we can get support from the source below:

<https://github.com/cswen-scut/chatroom> QT based text chatroom.

<https://github.com/kelecn/LAN-Chat-Room> QT implementation of LAN chat room, you can do chat, support different text color formats, support group chat/file transfer.

<https://github.com/haoxiuwang/webrtcTalk> Video call based on webrtc.

<https://github.com/iverxin/tcp_live> The opencv library collects images in real time and transmits them to the client using tcp.

BV1XS4y1T7iV Socket/communication

BV1aW4y1w7Ui TCP/communication

**Our work:**

**Overall Goal**

Considering the uncertainty of complexity, difficulty and limit of time, we have set our goal for two tier:

Tier 1: P2P connection, LAN connection.

Tier 2: Connection between more than 2 members, Internet connection.

**Overview of Implementation**

Our project, VidSync, will consist of three primary components:

**User Interface (UI)**: Developed using Qt Widgets, the UI will allow users to log in, manage a contact list, send text messages, and initiate video calls. We will implement a clean and intuitive interface, with the goal of minimizing user distractions while maintaining a focus on performance.

**Real-Time Communication (RTC)**: The core feature of our application is real-time communication. We will use Redis Pub/Sub to manage the real-time text chat between users and handle notification systems like message delivery and user presence (online/offline status). For video communication, we will leverage WebRTC for peer-to-peer video streaming, with Qt Multimedia handling the video capture, display, and control.

**Backend Server**: The server will manage user authentication, contact lists, and chat history using MySQL for persistent storage. The backend will also be responsible for handling WebSocket connections for message delivery and initiating WebRTC or Opencv signaling to set up peer-to-peer video connections.

**Detailed Architecture**

**Client (Qt-based):**

**Login/Register System**: Users will be able to log in and authenticate against the MySQL database.

**User Interface**: A responsive interface built with Qt, providing text chat, video call buttons, and notification systems for new messages or incoming calls.

**Multimedia Integration**: Using Qt Multimedia, users can access their camera and microphone, and the application will stream real-time audio and video to the recipient via WebRTC or Opencv.

**Server (Redis, MySQL, WebSocket):**

**User Management:** Store and manage user accounts, contact lists, and friend requests using MySQL.

**Redis Pub/Sub**: Handle real-time message relays between clients for the text chat.

Signaling Server: Manage WebRTC signaling messages to establish peer-to-peer video connections between clients.

**Project Highlights**

**Real-Time Performance**: Using Redis for fast message transmission ensures minimal latency in text chats. Peer-to-peer video connections using WebRTC will reduce server load and enhance video performance.

**Seamless Video Integration**: By integrating WebRTC and Qt Multimedia, we aim to provide a smooth and stable video chat experience.

**User-Friendly Interface**: A clean and modern interface designed with Qt, focusing on usability and performance.

**Lightweight and Efficient**: VidSync aims to deliver essential video chat features with minimal resource consumption, targeting both desktop and lightweight systems.

**Schedule:**

| Milestone | Description | Deadline |
| --- | --- | --- |
| Milestone 1: Project Setup | Establish Git repository, set up development environments, and define the basic architecture. | Oct 20, 2024 |
| Milestone 2: UI Design | Design the core user interface using Qt Widgets, including login and chat windows. | Oct 27, 2024 |
| Milestone 3: Backend Setup | Implement the MySQL database schema, user authentication, and Redis for real-time message handling. | Nov 10, 2024 |
| Milestone 4: Text Messaging | Integrate real-time messaging between clients using Redis Pub/Sub and WebSockets. | Nov 17, 2024 |
| Milestone 5: Video Calling | Implement peer-to-peer video calling using WebRTC and Qt Multimedia. | Nov 24, 2024 |
| Milestone 6: Testing and Debugging | Conduct testing, debugging, and optimizing performance of text chat and video calls. | Dec 4, 2024 |
| Milestone 7: Final Delivery | Complete final optimizations, prepare documentation, and submit the project. | Dec 7, 2024 |

**Team:**

**Wang Chuang** (Frontend/UI Developer): Responsible for designing and implementing the user interface using Qt. This includes login screens, chat windows, and the video call interface.

**Liu Changjun** (Frontend/UI Developer): Works alongside [Name 1] to enhance the user experience and handle UI responsiveness, ensuring a smooth interface across different devices.

**Wang Tianqing** (Backend Developer): Responsible for building the server-side logic, including user authentication, database integration with MySQL, and managing Redis for real-time messaging.

**Xu Lingfeng** (Backend Developer): Focuses on the WebRTC signaling server, ensuring video calls are successfully initiated between clients. This role also involves optimizing connection stability.

**Hong Kaiyang** (Multimedia Developer): Specializes in handling audio and video input/output using Qt Multimedia and integrating WebRTC for real-time video communication.

**Liu Zhijun** (Quality Assurance & Testing): Leads the testing effort, including unit tests, performance testing, and bug fixing. Also responsible for ensuring smooth collaboration between frontend and backend components.

Each team member will also contribute to code reviews and documentation.