IITB Summer Internship 2016

Project Report

NG I2I

Real Time Communication

with WebRTC

Principal Investigator

Prof. D.B. Phatak

Project In-Charge

Mr. Rahul Deshmukh

Project Mentors

Mr. Harish Satpute

Project Team Members

Mr. Omkar Damle

Mr. Rishabh Verma

Mr. Yash Trivedi

Summer Internship 2016 Project Approval Certificate

Department of Computer Science and Engineering

Indian Institute of Technology Bombay

The project entitled "NG I2I, Real Time Communication with WebRTC" submitted by Mr. Omkar Damle , Mr. Rishabh Verma and Mr. Yash Trivedi is approved for Summer Internship 2016 programme from 9th May 2016 to 1st July 2016, at Department of Computer Science and Engineering, IIT Bombay.

_____	_____
Prof. Deepak B. Phatak	Mr. Rahul Deshmukh
Dept of CSE, IITB	CDEEP, IITB
Principal Investigator	Project In-charge

Place: IIT Bombay, Mumbai

Date: _____

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Mr. Omkar Damle

Dhirubhai Ambani Institute of Information and Communication Technology, Gandhinagar

Mr. Rishabh Verma

National Institute of Technology, Silchar

Mr. Yash Trivedi

VIT University, Vellore

Date : _____

ACKNOWLEDGMENT

We, the summer interns of the group NG I2I, Real Time Communication with WebRTC, are overwhelmed in all humbleness and gratefulness to acknowledge our deep gratitude to all those who have helped us put our ideas to perfection and have assigned tasks well above the level of simplicity and into something concrete and We, whole heartedly thank **Prof. D.B. Phatak** for having faith in us, selecting us to be a part of his valuable project and for constantly motivating us to do better. We thank **Mr. Rahul Deshmukh** for providing us the opportunity to work on this project. We are also very thankful to our mentor **Mr. Harish Satpute** for his valuable suggestions. He was and is always there to show us the right track when needed help. With help of him brilliant guidance and encouragement, we all were able to complete our tasks properly and were up to the mark in all the tasks assigned. During the process, we got a chance to see the stronger side of our technical and nontechnical aspects and also strengthen our concepts. Last but not the least, we whole heartedly thank all our other colleagues working in different projects under **Prof. D.B Phatak** for helping us evolve better with their critical advice.

ABSTRACT

We have developed a Web Conferencing Application. The main technology used by our application is WebRTC. WebRTC is open source that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs. Our application supports Multi Peer video conferencing, Public chat, Private chat, Canvas drawing, Presentation sharing and Local video Sharing.

We have used the Kurento media server for streaming video content. Kurento is a WebRTC media server and a set of client APIs making simple the development of advanced video applications for WWW and smartphone platforms. A Java server has been used to establishing and handling connections between the Kurento Media Server and the clients. The chatting functionality is achieved through web sockets. For canvas, Fabric.js library has been used. Fabric.js is a powerful and simple Javascript HTML5 canvas library. The presentations are saved on the Java server and then transmitted to all the other clients. The Local video sharing enables us to share videos.

List of Figures

Contents

Chapter 1

Introduction

NG I2I, Real Time Communication with WebRTC is a suite of web-based real-time communication and sharing modules comprising of video conferencing, real-time whiteboard sharing, real-time presentation sharing and local video streaming. Our application works on all WebRTC compliant web browsers like Google Chrome, Mozilla Firefox and across all devices like desktop computers, laptops, tablets and mobile phones. It doesn't require the installation of any plugins or extensions.
It allows registered users to create or join existing meeting rooms. The users can view the video stream from all the other users and toggle between receiving both audio and video or only audio in case of poor bandwidth conditions. Users can chat with the all the other users of the meeting room using a public room chat and can chat privately with any other participant using a one-to-one private chat. It consists of a whiteboard allowing users to share various figures in real time. It also allows the users to present a presentation to all the other users in real time. It allows user to also share a locally stored video to other users. In addition to this, users can also be remotely muted by the meeting administrator. Our application defines three different access control modes – the administrator, the controller and the regular meeting attendees.

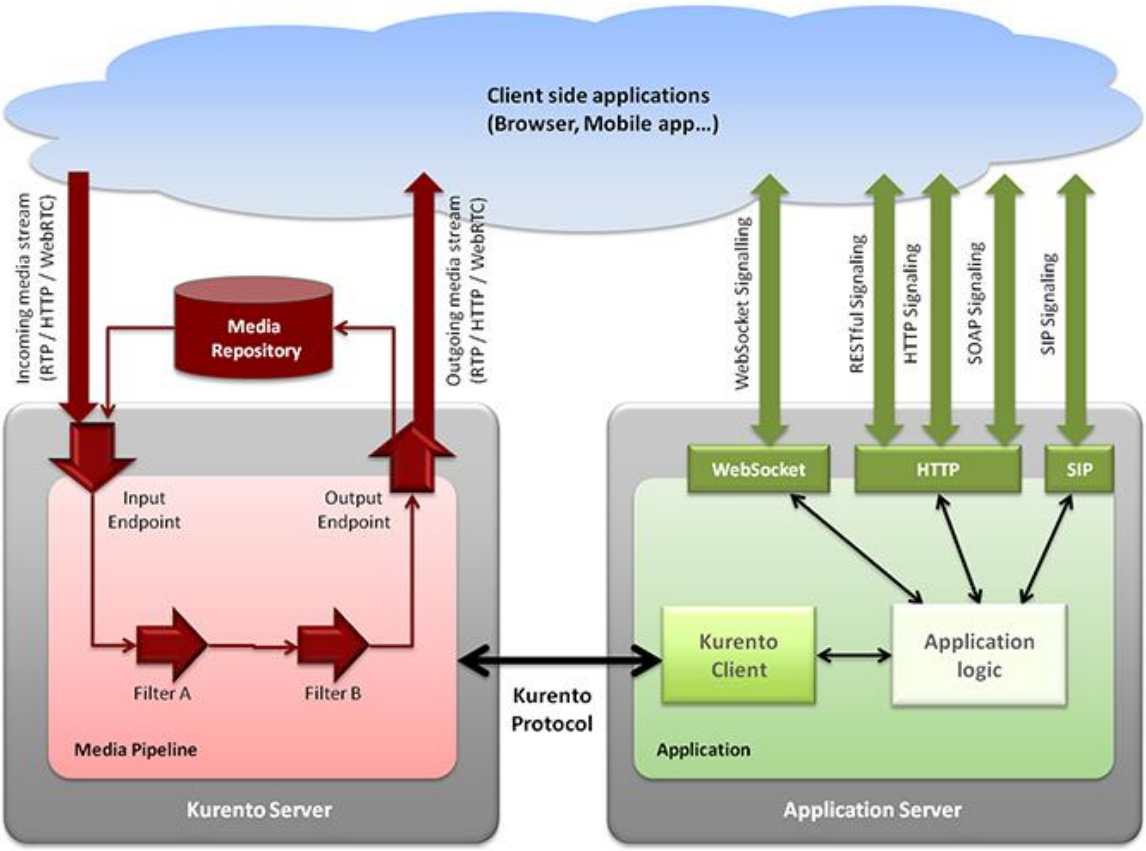

Figure 1.0

1.1 Purpose

In today's world everybody wants to remain connected in every way possible. Our prime aim was to make real time communication possible in video-audio mode and every other way in which we may increase the virtual interaction and communication. This was implemented using WebRTC and Kurento media server. Once the Audio-Video Communication is established ,our application ask for more features and we expand its reach to text chatting, whiteboard sharing, local video streaming and presentation sharing. These all features are being performed in real time using various web technologies

.

1.2 Scope

Our application has a very broad scope spanning the fields of business, distance learning, telecommuting / home offices, legal environments and telemedicine. It provides businesses with the ability to meet and to work with others over a distance. Teachers and students are able to see each other, share documents and discuss topics together in a situation similar to a traditional classroom setting. It helps users to save resources by meeting with clients and/or colleagues via videoconference. The use of videoconferencing technology is becoming more prevalent in today's courtrooms. Videoconferencing systems are used to enable testifying witnesses to appear in court without having to travel to the courtroom. A patient in a rural location can easily meet with a specialist from anywhere in the world.

1.3 Definitions and Abbreviations

1. WebRTC : Web Real-Time Communication
2. SDP : Session Description Protocol
3. AJAX : Asynchronous JavaScript and XML
4. RTD : Real Time Drawing
5. JS : JavaScript
6. CSS : Cascading Style Sheet

Chapter 2

Features and Modules

2.1 Features

1. **User Registry** – It maintains a list of registered users in our database and allows only those users to use our application.
2. **Dynamic Rooms** – The rooms are created dynamically giving users the freedom to name the rooms according to their choice.
3. **Time Elapsed** – The users are shown a clock which indicates the time elapsed since the beginning of the meeting.
4. **Participant Count** – The users are shown the number of participants currently in the meeting.
5. **Administrator** – The creator of the room is known as the administrator. He holds various powers like granting control to other users, muting other users and ending the meeting.
6. **Controller** – The user who has the rights to present a presentation, stream a local video or draw on the canvas is known as the controller.
7. **Public Chat** – It is a common per-room chat where the user can type a message which will be sent to all the members of the room.
8. **Private Chat** – It is a chat between any two users of the room which is exclusive to them.
9. **Videos Mode** – Displays the video stream from the cameras of all the other meeting participants.
10. **Canvas Mode** – Allows the current controller to add, manipulate and delete objects on the canvas which will be visible to all the other users.

11. **Presentation Mode** - Allows the current controller to present a presentation which will be visible to all the other users.
12. **Video Stream Mode** – Allows the current controller to stream a locally stored video file to al the other users in addition to his camera stream.
13. **Presenter Video** – While in the canvas drawing mode and the presentation mode, video of the current controller will be visible to all the users.
14. **Request Control** – Allows user to request the administrator to become controller.
15. **Mute User** – Allows the administrator to mute a given user for all the meeting participants.
16. **Toggle Audio/Video** – Allows each user to decide whether to receive audio or video from a given meeting participant.

2.2 Real Time Canvas - Whiteboard Sharing

The canvas is an essential feature of the application. It enables the presenter to express his ideas on the whiteboard. The presenter can draw on canvas and edit the objects already drawn. The video of the presenter is shown in the top right corner. Any participant can download the canvas in png format and save for future reference. The canvas can also be sent to the server side for storing.

If any participant joins late , then all objects on the canvas are sent to him and he/she can continue with the other participants in the conference.

All the features of the canvas are explained in the toolbar section.

2.2.1 Toolbar

The toolbar implements the following basic drawing and editing options for the canvas module. The toolbar has two modes -

a. Draw mode -

In this mode the presenter can :

- draw/erase
- add shapes/images to the canvas.

b. Edit mode -

In this mode the presenter can :

- select objects and move them around.
- change stroke and fill of objects
- resize/rotate the objects.
- bring object to front or send back.
- delete the object.

Apart from these in both modes, the user can undo, redo addition of objects ,download canvas , clear canvas and send the canvas to server side. The toolbar icons are explained hereafter :

1. Stroke color - This color is used to change the stroke of shapes and also is the color used by the free drawing pencil.
2. Fill color - This color is used to change the fill of shapes and also is the color used by the text box.
3. Text - A text box for adding text in the canvas.
4. Line and shapes - insert straight lines or shapes such as circle, rectangle, triangle. The width of the line can be adjusted by the slider. The size of the line and shapes can be dynamically changed while drawing itself. The stroke and fill color can be changed in edit mode.
5. Pencil and RTD - A free drawing tool. The RTD (Real time drawing) feature allows sending the drawing real time.

6. Eraser - Basic implementation of an eraser. The width can be changed dynamically.
7. Undo and Redo - The addition of objects can be undone and redone.
8. Clear canvas - Clears the entire canvas. Cannot be undone.
9. Download - downloads the entire canvas as a png image.
10. Image - Adds an image to the canvas. The image can be chosen from the user's folders.
11. Up and down - Used to send objects backward or to bring objects to front on the canvas.

2.2.2 Canvas Board

We have a fix sized canvas board on which all the features listed under toolbox section are implemented.

2.3 Real Time Presentation Sharing

The Application developed also has a feature to share the presentation , each user can upload its own presentation in pptx format which then uploaded to server and broadcasted to all users present in the room. Only the user who is currently the Controller can access the modifying controls of this module. All Other Participating users even the Admin can only view the presentation shared by controller. Only controller can change the slide on his/her own convenience.

Once the controller chooses to switch to presentation sharing mode each and every user is brought to the same module.

Presenter's Video :The Video of the presenter/controller is shown to all the users present in the meeting on the top right corner of the Application .It is dynamically changed on change of Controller. The Video is extracted from the WebRTC connection made hence no other stream is required from the presenter to all other users.

Asynchronous Uploading: The Controller is given a Form to upload his own one pptx file to be uploaded to the server via form given under Presentation module. On form submission an AJAX call is made to server to upload the file and as AJAX call is used in Asynchronous mode hence allowing controller to perform any other operation such as canvas drawing. The controller will then be notified by server on upload completion or on error.

Start Presentation: Even after uploading the presentation if the controller want to wait for some more time before starting the presentation then he may do this as the presentation is broadcasted to all on click of start button.

Controls for Presentation: Once the presentation is started only controller has the control over navigation of the presentation and can bring all user to any slide he wants by using the controls button provided.

Animation: The Presentation Slide are animated for all the users on change of the slide by the controller.

2.4 Local Video Streaming and Sharing

The Application developed also has a feature to stream the local video file , each user can upload its own video file which then uploaded to server and broadcasted to all users present in the room. Only the user who is currently the Controller can access the modifying controls of this module. All Other Participating users even the Admin can only view the video shared by controller.

Once the controller chooses to switch to video streaming mode each and every user is brought to the same module.

Controller’s Video :The Video of the controller is shown to all the users present in the meeting on the top right corner of the Application .It is dynamically changed on change of Controller. The Video is extracted from the WebRTC connection made hence no other stream is required from the presenter to all other users.

Asynchronous Uploading: The Controller is given a Form to upload his own one video file to be uploaded to the server via form given under video streaming module. On form submission an AJAX call is made to server to upload the file and as AJAX call is used in Asynchronous mode hence allowing controller to perform any other operation such as presentation sharing. The controller will then be notified by server on upload completion or on error.

Start Video: Even after uploading the video if the controller want to wait for some more time before starting the presentation then he may do this as the video is broadcasted to all on click of start button.

Controls for Video: Once the video is started only controller has the control over pause/play of the video and can do this using the controls button provided.

2.5 Video Conferencing

Our main Audio/Video Conferencing is shown when user login and can see the video of other and chat with them privately in this module , this module open on click of Video Mode present on Menu Bar.

Any video can be enlarged on click. The name tag is shwon with each video and other controls for Admin.

2.6 Participant List and features

The list of all the connected participants is shown on the right side of the screen. There are two buttons alongside each user.

1. Chat - Clicking on this button will enable the user to have a private chat with that particular participant.
2. Get Audio/video - Clicking on this button will get the audio only/video for that particular participant.

This list is responsive as well as interactive. If the number of participants increases, the user can scroll over the list.

2.7 Room Chat

This is a feature which allows all the participants in the room to chat together. If a user enters the the meeting late, the previous messages can still be seen by him. If a user wants to become the presenter, he/she can click on the request control button and then the message will be sent in the room chat.

Chapter 3

Technologies Used

In this chapter all the technologies which are extensively used in development are listed and described in brief.

3.1 JavaScript

JavaScript is a high-level, dynamic, untyped, and interpreted programming language. It is employed by a majority of websites and it is supported by all modern Web browsers without plugins. JavaScript supports object-oriented, imperative, and functional programming styles. It has an API for working with text, arrays, dates and regular expressions.

Reasons for using JavaScript –

Speed - Being client-side, JavaScript is very fast because any code functions can be run immediately instead of having to contact the server and wait for an answer.

Server Load - Being client-side reduces the demand on the website server.

Increased Interactivity − Interfaces that react when the user hovers over them with a mouse or activates them via the keyboard can be created.

3.2 AJAX

AJAX (Asynchronous JavaScript and XML) is a set of web development techniques using many web technologies on the client-side to create asynchronous Web applications. Previously, each time the browser reloaded a page because of a partial change, all of the content had to be re-sent, even though only some of the information had changed. This

placed additional load on the server and made bandwidth the limiting factor on performance.

With AJAX, web applications can send data to and retrieve from a server asynchronously without interfering with the display and behavior of the existing page.

Reasons for using AJAX –

- **Better interactivity** - AJAX allows easier and quicker interaction between user and website as pages are not reloaded for content to be displayed.
- **Compact** - With AJAX, several multi-purpose applications and features can be handled using a single web page, avoiding the need for clutter with several web pages.

3.3 WebRTC

WebRTC is an open framework for the web that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities without the need of either internal or external plugins. It includes the fundamental building blocks for high-quality communications on the web, such as network, audio and video components used in voice and video chat applications.

WebRTC enables all kinds of real time communication such as audio, video and text between users by utilizing the browsers. It is a free, high-quality, complete solution available that enables communication in the browser.

It includes and abstracts key NAT and firewall traversal technology, using STUN, ICE, TURN, RTP-over-TCP and support for proxies.

Reasons for using WebRTC –

Ease of use - Real-time communication is supported without the need for additional applications or plug-ins. Downloading, installing and updating plugins can be complex, error prone and annoying.

Security - WebRTC enforces the usage of encryption for the media. Thereby, WebRTC provides a higher security level than most currently available commercial telephony systems.

3.4 Web Sockets

Modern web applications require more interactivity than ever before for client/server communications. HTTP, however, wasn't built to deliver the kind of interactivity needed today. "Push" or Comet techniques, such as long-polling, emerged as a way to allow a server to push data to a browser. Because these techniques usually rely on HTTP, they present some disadvantages for client/server communications, such as HTTP overhead. These disadvantages result in less efficient communication between the server and the web browser, especially for real-time applications.

WebSocket provides an alternative to this limitation by providing bi-directional, full-duplex, real-time, client/server communications. The server can send data to the client at any time.. WebSocket also provides greater scalability for message-intensive applications because only one connection per client is used (whereas HTTP creates one request per message).

The life cycle of a WebSocket is easy to understand as well:

Client sends the Server a handshake request in the form of a HTTP upgrade header with data about the WebSocket it’s attempting to connect to.

The Server responds to the request with another HTTP header, this is the last time a HTTP header gets used in the WebSocket connection. If the handshake was successful, the server sends a HTTP header telling the client it’s switching to the WebSocket protocol.

Now a constant connection is opened and the client and server can send any number of messages to each other until the connection is closed. These messages only have about 2 bytes of overhead.

The WebSocket API was introduced with Java EE7.

3.5 Apache POI

Apache POI library component- XSLF has been used for accessing the PPT on the JAVA Server Side on Spring Boot Application.

Apache POI is a popular API that allows programmers to create, modify, and display MS Office files using Java programs. It is an open source library developed and distributed by Apache Software Foundation to design or modify Microsoft Office files using Java program. It contains classes and methods to decode the user input data or a file into MS Office documents.

XSLF is the POI Project's pure Java implementation of the PowerPoint 2007 OOXML (.xlsx) file format.

3.6 Kurento

Kurento is a WebRTC media server and a set of client APIs enabling the development of advanced video applications for WWW and smartphone platforms. Kurento Media Server features include group communications, transcoding, recording, mixing, broadcasting and routing of audiovisual flows. It also provides advanced media processing capabilities involving computer vision, video indexing, augmented reality and speech analysis. Its modular architecture makes simple the integration of third party media processing algorithms (i.e. speech recognition, sentiment analysis, face recognition, etc.)

Kurento's core element is Kurento Media Server, responsible for media transmission, processing, loading and recording. It has been implemented in low level technologies based on GStreamer to optimize the resource consumption. It provides the following features:

Networked streaming protocols, including HTTP, RTP and WebRTC.

Group communications (MCUs and SFUs functionality) supporting both media mixing and media routing/dispatching.

Media storage supporting writing operations for WebM and MP4 and playing in all formats supported by GStreamer.

Kurento Media Server capabilities are exposed by the Kurento API to application developers. This API is implemented by means of libraries called Kurento Clients. Kurento offers two clients for Java and JavaScript.

Kurento is based on two concepts that act as building blocks:

3.6.1 Media Elements

A Media element is a functional unit performing a specific action on a media stream. Media Elements are a way every capability is represented to the application developer. Media elements are capable of receiving media from other elements and of sending media to other elements. Depending on their function, media elements can be split into different groups:

Input Endpoints: Media elements capable of receiving media and injecting it into a pipeline. There are several types of input endpoints. File input endpoints take the media from a file, Network input endpoints take the media from the network, and Capture input endpoints are capable of capturing the media stream directly from a camera or other kind of hardware resource.

Filters: Media elements in charge of transforming or analyzing media. Hence there are filters for performing operations such as mixing, analyzing, augmenting, etc.

Hubs: Media Objects in charge of managing multiple media flows in a pipeline. A Hub has several hub ports where other media elements are connected. Depending on the Hub type, there are different ways to control the media. For example, there are a Hub called Composite that merge all input video streams in a unique output video stream with all inputs in a grid.

Output Endpoints: Media elements capable of taking a media stream out of the pipeline. Again, there are several types of output endpoints specialized in files, network, screen, etc.

3.6.2 Media PipeLine

A Media Pipeline is a chain of media elements, where the output stream generated by one element (source) is fed into one or more other elements input streams (sinks). Hence, the pipeline represents a machine capable of performing a sequence of operations over a stream.

3.7 Session Description Protocol

The Session Description Protocol (SDP) is a format for describing streaming media initialization parameters. Both parties of a streaming media session exchange SDP files to negotiate and agree in the parameters to be used for the streaming.

SDP is intended for describing multimedia communication sessions for the purposes of session announcement, session invitation, and parameter negotiation. SDP does not deliver media itself but is used between end points for negotiation of media type, format,

and all associated properties. The set of properties and parameters are often called a session profile. SDP is designed to be extensible to support new media types and formats.

3.8 Material Design and W3.CSS

W3.CSS is a Cascading Style Sheet (CSS) developed by w3schools.com. It helps in creating faster, beautiful, and responsive websites. It is inspired from Google Material Design.

Some of its salient features are as follows:

- In-built responsive designing
- Standard CSS
- Inspired by Google Material Design
- Free to use

3.9 Responsive Design

- W3.CSS has in-built responsive designing so that the website created using W3.CSS will redesign itself as per the device size.
- W3.CSS has a 12 column mobile-first fluid grid that supports responsive classes for small, large, and medium screen sizes.
- W3.CSS classes are created in such a way that the website can fit any screen size.
- The websites created using W3.CSS are fully compatible with PC, tablets, and mobile devices.
- It supports paper-like design.
- It supports shadows and bold colors.
- The colors and shades remain uniform across various platforms and devices.

3.10 Kurento Client APIs

Kurento API is an object oriented API to create media pipelines to control media. It can be seen as an interface to Kurento Media Server. It can be used from the Kurento Protocol or from Kurento Clients.

A **Kurento Client** is a programming library (Java or JavaScript) used to control **Kurento Media Server** from an application. For example, with this library, any developer can create a web application that uses Kurento Media Server to receive audio and video from the user web browser, process it and send it back again over Internet. Kurento Client exposes the Kurento API to developers. In the Application developed Kurento JAVA API is used.

Kurento Utils JS Library

Kurento Utils is a wrapper object of an RTCPeerConnection . This object is aimed to simplify the development of WebRTC-based applications.

The library offers a WebRTCPeer object, which is a wrapper of the browser's RTCPeerConnection API. Peer connections can be of different types: unidirectional (send or receive only) or bidirectional (send and receive).

3.11 Fabric.js

Fabric.js is a powerful Javascript library that makes working with HTML5 canvas a breeze. Fabric provides a missing object model for canvas, as well as an SVG parser, layer of interactivity, and a whole suite of other indispensable tools. It is a fully open-source project, licensed under MIT, with many contributions over the years.

Canvas allows us to create some absolutely amazing graphics on the web these days. But the API it provides is disappointingly low-level. It's one thing if we simply want to draw few basic shapes on canvas and forget about them. But as soon as there's need for any kind of interaction, change of picture at any point, or drawing of more complex shapes — situation changes dramatically.

Native canvas methods only allow us to fire off simple graphic commands, blindly modifying entire canvas bitmap. Want to draw a rectangle? Use fillRect(left, top, width, height). Want to draw a line? Use a combination of moveTo(left, top) and lineTo(x, y). It's as if we're painting canvas with a brush, layering more and more oil on top, with very little control.

Instead of operating on such low level, Fabric provides simple but powerful object model on top of native methods. It takes care of canvas state and rendering, and lets us work with “objects” directly.

3.12 Spring Boot Application

Spring Boot makes it easy to create stand-alone, production-grade Spring based Applications can “just run”.It takes an opinionated view of the Spring platform and third-party libraries so we can get started with minimum fuss. Most Spring Boot applications need very little Spring configuration.

Spring Boot is well suited for web application development. We can easily create a self-contained HTTP server using embedded Tomcat, Jetty, or Undertow. Most web applications will use the spring-boot-starter-web module to get up and running quickly. The SpringApplication class provides a convenient way to bootstrap a Spring application that will be started from a main() method. In many situations you can just delegate to the static SpringApplication.run method:

We can use Spring Boot to create Java applications that can be started using java -jar or more traditional war deployments. It also provide a command line tool that runs “spring scripts”.

Its primary goals are:

- Provide a radically faster and widely accessible getting started experience for all Spring development.

- Be opinionated out of the box, but get out of the way quickly as requirements start to diverge from the defaults.
- Provide a range of non-functional features that are common to large classes of projects (e.g. embedded servers, security, metrics, health checks, externalized configuration).
- Absolutely no code generation and no requirement for XML configuration

Chapter 4

Design

4.1 Use Case

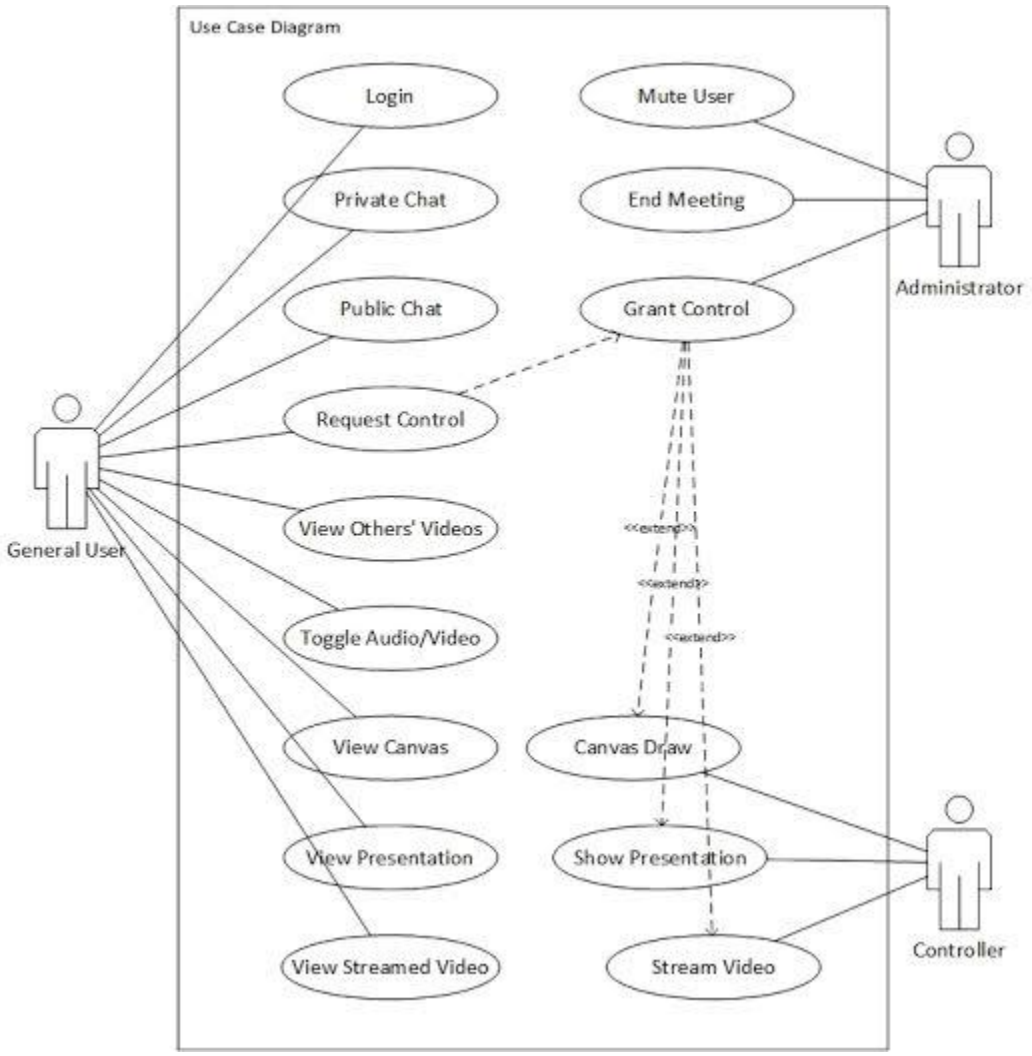

Figure 4.1

4.2 Class Diagram

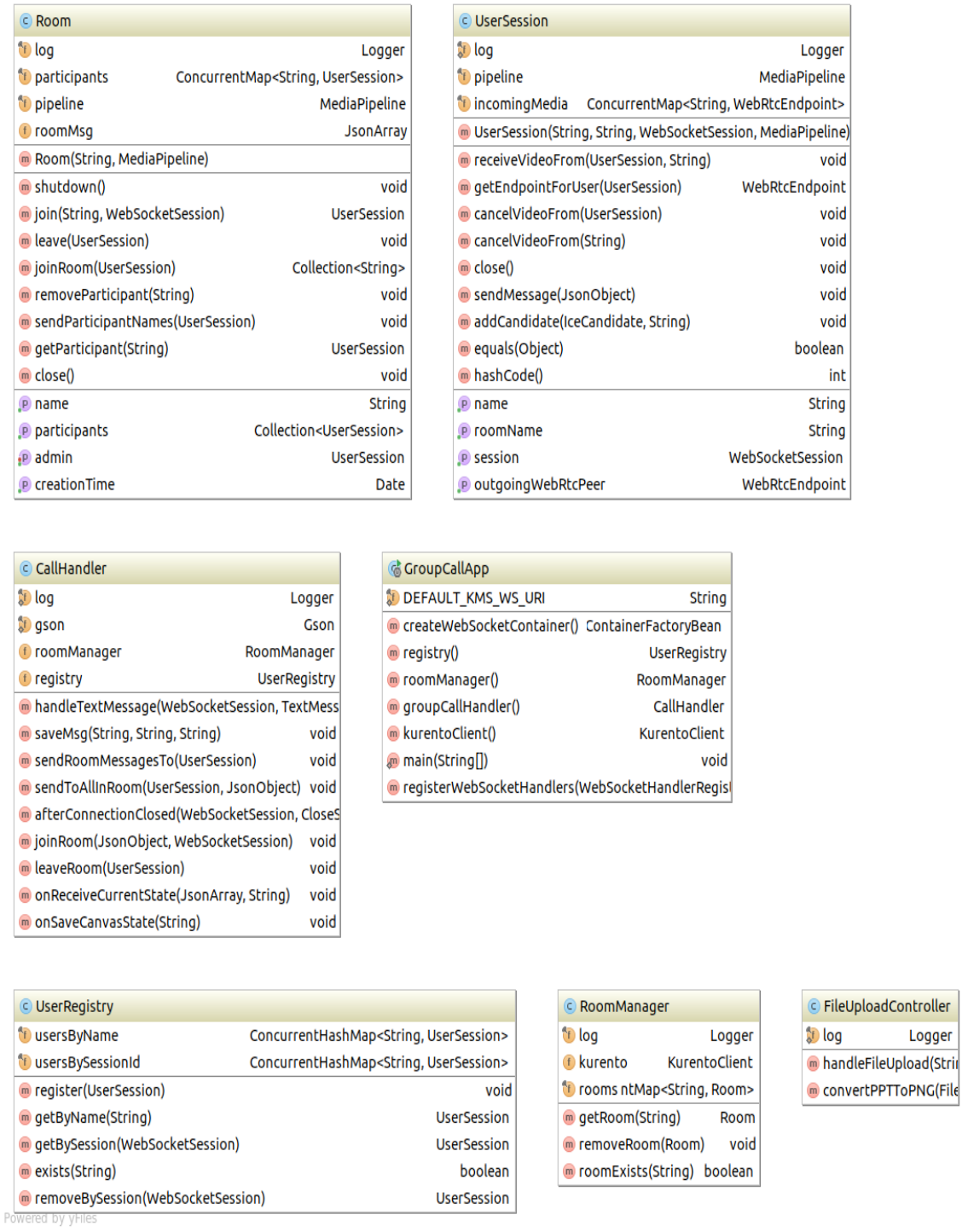

Figure 4.2

4.3 InDepth Signaling diagram

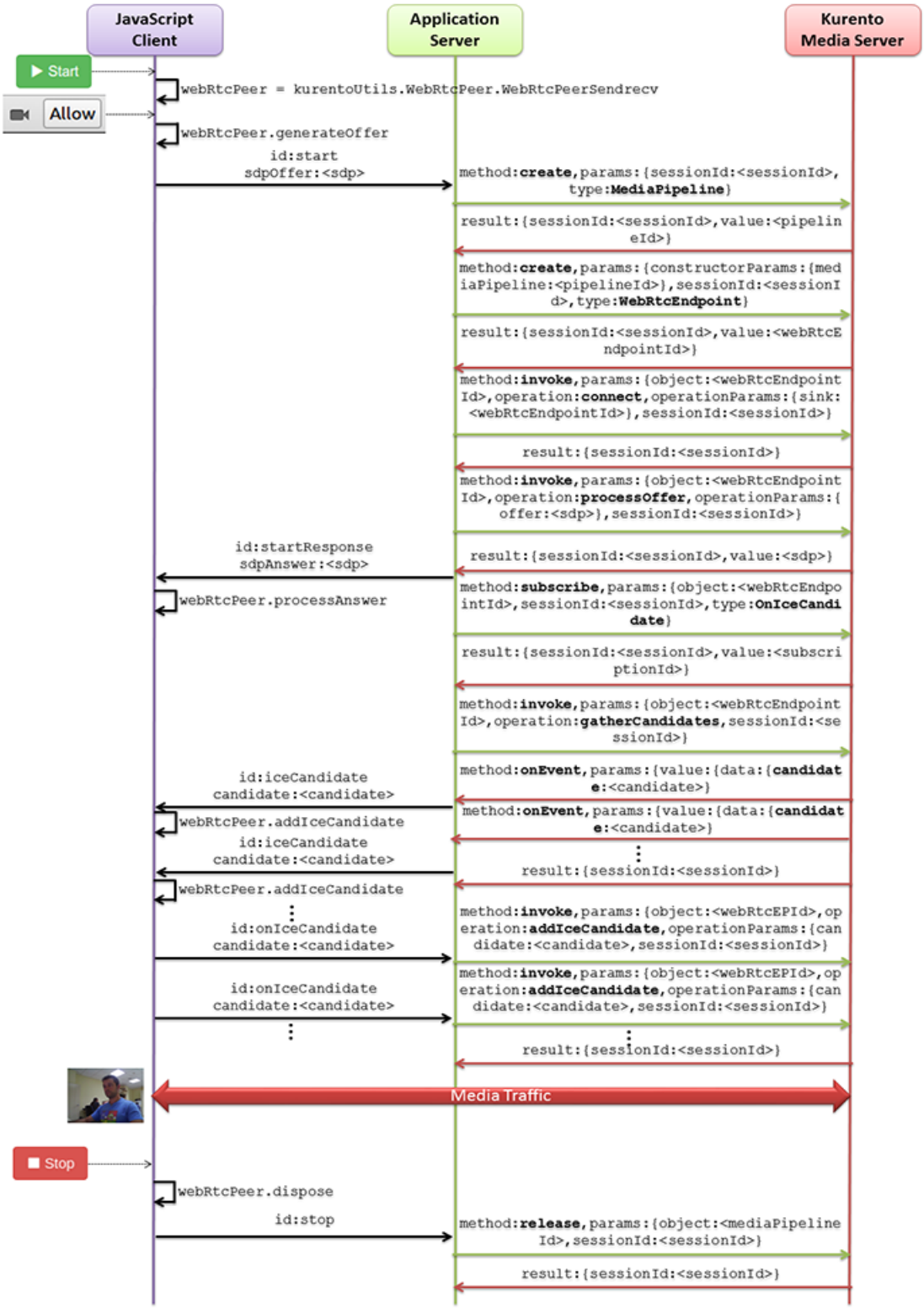

Figure 4.3

4.4 Levels of Control Division and Distribution

4.4.1 Level I - General User

This Level possesses the Least Controls over the Application and all the controls given to this level user cannot affect any other user present in the Room. Every Participant on joining the Room possess this level of control, except Admin(Level III).

Controls Given:

1. Can Only Request Admin to become Controller (Level II).
2. Can view Canvas.
3. Can view Presentation.
4. Can view Locally Shared Video.
5. Can Text Chat Privately with any User.
6. Can send message to all users using Public Chat dialog.
7. Get Video and Audio of all other users in the Meeting.
8. Has Control to turn Off Video of a User (say B) to him(say A), that is switching to Audio Only mode, this will not affect any user B to receive the A's video.

4.4.2 Level II - Controller

This Level possesses Controls over the Application to modify the Data broadcast. Every Participant of Level I can be promoted to Level II by admin on Request . By default Admin(Level III) is the Controller. A meeting can have only one Controller at a time.

Controls Given:

1. Can Draw on Canvas continuing along with the current state of the Canvas.
2. Can Upload its own presentation to be presented to all users present in the Meeting.
3. Can share its own video to be streamed to all users present in the Meeting.

4. Possess all the Controls of Level I user also.

4.4.3 Level III - Administrator

This level is the supreme level , the user creating the Room is by Default Admin. Initially Admin is also the Controller. This Level controls can affect other user present in the Room.

Controls given:

1. Can promote anyone present in the Meeting from Level I to Level II.
2. Can withdraw anytime the control of Level II from any user.
3. Can ignore the Request made by user.
4. Can mute a user for all others in the Meeting.
5. Unmute the muted user .
6. It is the necessary entity for a meeting , as on admin leave meeting is automatically closed.

Result and Conclusion

Under the Real Time Communication with WebRTC we have successfully implemented a suite of web-based real time communication and sharing modules. The users are able to create or join a room. They are able to receive audio and video streams from all other users. Users can draw on the whiteboard, share their presentations and locally stored videos with other users. They are also able to communicate with each other using private or public chat. Our application has been tested successfully on laptops, tablets and mobile phones.

Thus we have successfully implemented and completed our project and the assigned tasks.

Future Work

Our application uses kurento media server for audio and video communication , but for file sharing and chat message we are still dependent on WebSockets. So in future work we may switch to DataChannel API provided by Kurento media server.

The basic functionalities have been implemented in canvas like drawing shapes/images and editing them. Now some more features can be added like different brushes, more shapes, changing font size and style,etc.

Currently only .pptx presentations can be uploaded. In future other formats like .ppt and .odt should also be supported.

Testing of the application on a commercial level must be done for getting in depth limitation and future enhancements of the Application.

References

1. Fabric js - http://fabricjs.com/docs/fabric.Canvas.html
2. The noun project icons -https://thenounproject.com/
3. Presentation ppt to png images- http://www.tutorialspoint.com/apache_poi_ppt/apache_poi_ppt_to_image.htm
4. CSS styling - http://www.w3schools.com/w3css/
5. Kurento official site -https://www.kurento.org/
6. www.stackoverflow.com
7. www.html5rocks.com/en/tutorials/webrtc/basics/
8. https://webrtc.org/