External Project Report on Computer Networking (CSE3034)

Intranet based Video over Wi-Fi (VoWi-Fi) system

Submitted by

Manshika Gerg	**2141007007**
Jashmeet Panda	**2141016056**
Tushar Prasad	**2141016321**
Nandita Singh	**2141019333**
Aman Kumar Singh	**2141019334**

B. Tech. CSE 5th Semester (Section I)

INSTITUTE OF TECHNICAL EDUCATION AND RESEARCH
(FACULTY OF ENGINEERING)
SIKSHA 'O' ANUSANDHAN (DEEMED TO BE UNIVERSITY), BHUBANESWAR, ODISHA

Declaration

We, the undersigned students of B. Tech. of **Computer Science and Engineering** Department hereby declare that we own the full responsibility for the information, results etc. provided in this PROJECT titled **Intranet based Video over Wi-Fi (VoWi-Fi) system** submitted to **Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar** for the partial fulfillment of the subject **Computer Networking (CSE 3034)**. We have taken care in all respect to honor the intellectual property right and have acknowledged the contribution of othersfor using them in academic purpose and further declare that in case of any violation of intellectual property right or copyright we, as the candidate(s), will be fully responsible for the same.

Manshika Gerg

2141007007

Jashmeet Panda

2141016056

Tushar Prasad

2141016321

Nandita Singh

2141019333

Aman Kumar Singh

2141019334

DATE – 11th January 2024

PLACE –

Abstract

The Intranet-based Video over Wi-Fi (VoWi-Fi) project aims to revolutionize internal communication within organizations by implementing a cutting-edge video streaming system.

Leveraging the ubiquity of Wi-Fi networks, this system facilitates seamless video communication, fostering collaboration and information sharing among employees. The project focuses on optimizing the Intranet infrastructure to support high-quality, low-latency video streaming, ensuring a smooth and immersive user experience.

Key features include adaptive streaming protocols for varying network conditions, robust security measures to protect sensitive information, and compatibility with a range of devices.

The VoWi-Fi system not only enhances traditional communication channels but also introduces novel ways to connect, such as video conferencing, live streaming of presentations, and on-demand video content delivery.

This project addresses the growing need for flexible and efficient communication tools within corporate environments, ultimately promoting productivity and collaboration among team members.

Contents

1. Introduction

In the ever-evolving landscape of corporate communication, the integration of advanced technologies has become imperative for enhancing connectivity and collaboration within organizations. One such groundbreaking solution that has revolutionized internal communication is the Intranet-based Video over Wi-Fi (VoWi-Fi) system. This innovative system leverages the power of local networks to seamlessly transmit video content, fostering a dynamic and interactive environment within the confines of an organization.

Gone are the days of traditional communication methods; the Intranet-based VoWi-Fi system represents a paradigm shift by combining the ubiquity of Wi-Fi connectivity with the visual richness of video communication. This amalgamation not only transcends geographical barriers but also augments the efficiency and effectiveness of internal communication channels. As we delve into the intricacies of this cutting-edge system, we will explore how it empowers enterprises to engage in real-time video interactions, streamline information dissemination, and ultimately cultivate a more connected and collaborative workplace. The Intranet-based VoWi-Fi system is not merely a technological advancement; it is a catalyst for a new era of corporate communication.

The client-server webcam streaming application is designed to enable real-time video transmission from a server to multiple clients over a network. The primary objective of this project is to facilitate remote viewing of webcam-captured video feeds across connected devices.

2. Problem Statement

Implement an **Intranet based Video over Wi-Fi (VoWi-Fi) system** within a single access point using socket programming. Demonstrate how to capture video frames from a camera on the server side, transmit them over a local network using sockets, and display them on the client side. (use Java Programming)

Features:

- **Server-Client Architecture**: Establish a server-client architecture, where the server captures video frames from a camera and sends them to the client for display.
- **Video Capture and Transmission**: Perform video capture, processing, and transmission over a local network.
- **Socket Communication:** Use basic socket programming to transmit video frame data between the client and server over a specified port.
- **Real-time Video Display**: The client side processes and displays the received video frames in real-time, demonstrating the continuous transmission of video data.

3. Methodology

VIDEO CLIENT PSEUDOCODE

1. Connect to the server using Socket and specific IP address and port
2. Initialize ObjectInputStream to receive data from the server
3. Create GUI for displaying the streamed content
4. Loop:
   5. Receive ImageIcon from server using ObjectInputStream.readObject()
   6. Update GUI with the received ImageIcon
7. Close resources (Socket, etc.) when done

VIDEO CLIENT ALGORITHM

Client Algorithm:

1. Connect to Server:
   - Create a Socket and connect to the server using the server's IP address and port.
2. Initialize ObjectInputStream:
   - Create an ObjectInputStream to receive data from the server.
3. Create GUI for Display:
   - Set up a graphical user interface (GUI) for displaying the streamed content.
4. Receiving Loop:
   - Enter an infinite loop.
     - Receive an ImageIcon from the server using ObjectInputStream.readObject().
     - Update the GUI with the received ImageIcon.
5. Close Resources:
   - Close the Socket and any other open resources when done receiving.

VIDEO SERVER PSEUDOCODE

1. Initialize Webcam.
2. Open Webcam and Set View Size.
3. Initialize ServerSocket on a specific port.
4. Wait and Accept a client connection using ServerSocket.accept()
5. Initialize ObjectOutputStream to send data to the client.
6. Create and Show GUI for streaming (optional).
7. Loop:
   8. Capture frame from Webcam
   9. Convert frame to ImageIcon
   10. Send ImageIcon to client using ObjectOutputStream.writeObject()
   11. Flush the ObjectOutputStream
   12. Pause for a short duration (e.g., Thread.sleep()).
13. Close resources (Webcam, ServerSocket, etc.) when done.

VIDEO SERVER ALGORITHM

1.Initialize Webcam:
- Create an instance of the webcam object.

2.Open Webcam and Set View Size:
- Open the webcam.
- Set the desired view size for capturing frames.

3.Initialize ServerSocket:
- Create a ServerSocket object on a specified port.

4.Accept Client Connection:
- Wait for and accept a client connection using ServerSocket.accept().

5.Initialize ObjectOutputStream:
- Create an ObjectOutputStream to send data to the client.

6.Create GUI for Streaming (Optional):
- Optionally, set up a graphical user interface (GUI) for streaming the webcam content.

7.Streaming Loop:
- Enter an infinite loop.

*Capture a frame from the webcam.

*Convert the frame to an ImageIcon.

*Send the ImageIcon to the client using ObjectOutputStream.writeObject().

*Flush the ObjectOutputStream.

*Pause for a short duration (e.g., Thread.sleep()).

8.Close Resources:

- Close the webcam, ServerSocket, and any other open resources when done streaming.

4. Implementation

CLIENT CODE

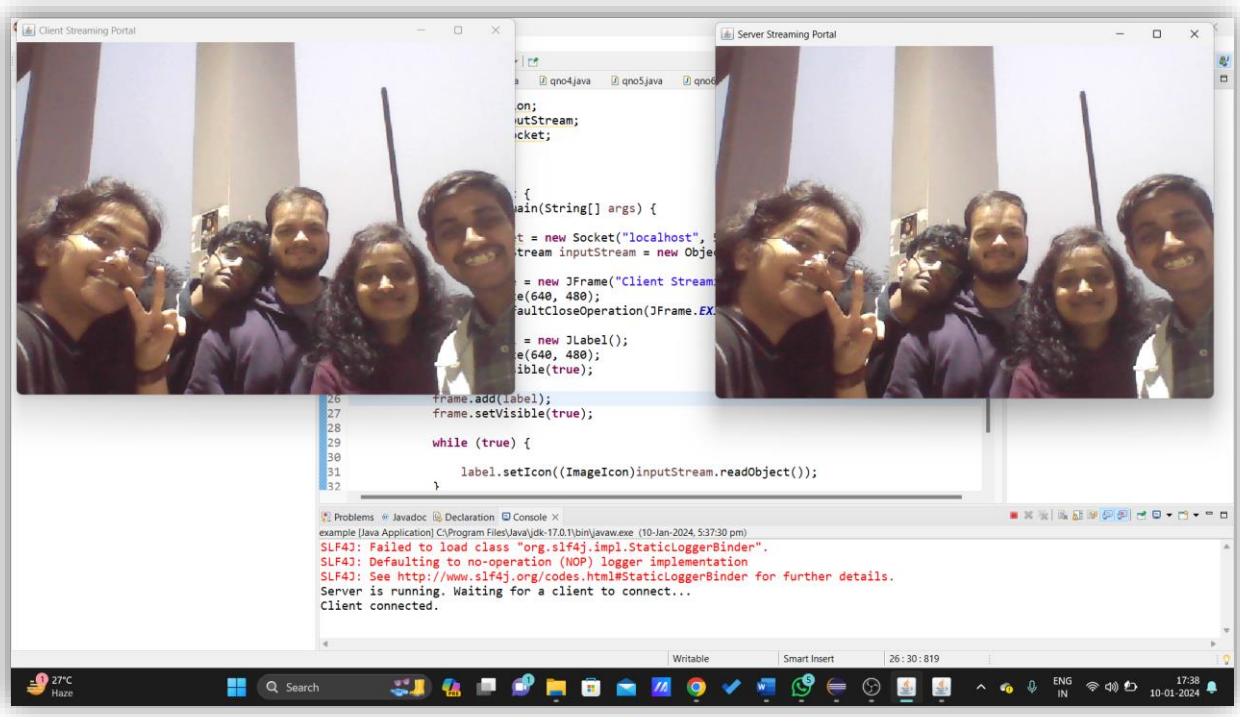
```
import javax.swing.*;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.net.Socket;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.net.ServerSocket;
import java.net.Socket;


public class client2 {
   public static void main(String[] args) {
      try {
         Socket socket = new Socket("localhost", 5555);
         ObjectInputStream inputStream = new ObjectInputStream(socket.getInputStream());

         JFrame frame = new JFrame("Client Streaming Portal");
         frame.setSize(640, 480);
         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

         JLabel label = new JLabel();
         label.setSize(640, 480);
         label.setVisible(true);

         frame.add(label);
         frame.setVisible(true);

         while (true) {

                label.setIcon((ImageIcon)inputStream.readObject());
         }
      } catch (IOException | ClassNotFoundException e) {
         e.printStackTrace();
      }
   }}
```

SERVER CODE

```
import com.github.sarxos.webcam.Webcam;

import java.io.IOException;
import java.io.ObjectOutputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.awt.Dimension;
import java.awt.image.BufferedImage;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;

public class server2 {
   public static void main(String[] args) {
      try {
         Webcam webcam = Webcam.getDefault();
         webcam.setViewSize(new Dimension(640, 480));
         webcam.open();

         ServerSocket serverSocket = new ServerSocket(5555);
         System.out.println("Server is running. Waiting for a client to connect...");

         Socket clientSocket = serverSocket.accept();
         System.out.println("Client connected.");

        ObjectOutputStream outputStream = new ObjectOutputStream(clientSocket.getOutputStream());
         JFrame frame = new JFrame("Server Streaming Portal");
         frame.setSize(640, 480);
         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

         JLabel label = new JLabel();
         label.setSize(640, 480);
         label.setVisible(true);

         frame.add(label);
         frame.setVisible(true);
         while (true) {
            BufferedImage image = webcam.getImage();
            ImageIcon imageIcon = new ImageIcon(image);
            label.setIcon(imageIcon);
            outputStream.writeObject(imageIcon);
            outputStream.flush();


            Thread.sleep(100);
```

```
            }
        } catch (IOException | InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

5. Results & Interpretation



When we run the server side first and the client side second, two portals are opened - the **Client Streaming Portal** (on the left) and the **Server Streaming Portal** (on the right). We can then watch a real-time video display from both ends with zero delay.

6. Conclusion

In conclusion, the Intranet-based Video over Wi-Fi (VoWi-Fi) system stands as a remarkable technological advancement with significant implications for communication within organizational settings. This project seamlessly integrates video communication over existing Wi-Fi infrastructure, fostering efficient and dynamic collaboration among team members. By leveraging the power of Intranet, the system not only ensures secure and reliable video transmission but also enhances the overall connectivity experience.

The implementation of VoWi-Fi not only reduces the dependence on external communication platforms but also provides a cost-effective solution for organizations. Its user-friendly interface and adaptability contribute to a streamlined workflow, facilitating enhanced productivity and communication efficiency. Furthermore, the system's scalability and flexibility make it adaptable to diverse organizational structures.

As we navigate the ever-evolving landscape of digital communication, the Intranet-based VoWi-Fi system emerges as a cutting-edge solution, aligning seamlessly with the demands of modern workplaces. The successful execution of this project signifies a step forward in harnessing technology to optimize internal communication, fostering a more connected and collaborative work environment.

References

(as per the IEEE recommendations)

[1] Computer Networks, Andrew S. Tannenbaum, Pearson India.

[2] Java Network Programming by Harold, O'Reilly (Shroff Publishers).

[3] GeeksforGeeks