SOC-App : Secure Online Connection

Tracing IP anomalies using machine learning with automated NSFW filter whilst implementing solution for unique user identification

11.02.2022

Proposed By- Team Code-Hashiras

Members-

- Dev Pant
- Ekta
- Amardeep Saha
- Prasoon
- Rishav Mazumdar
- Aman Kumar

Introduction to Problem

The video conferencing market is a multi-billion dollar industry that has been meeting with constant demand of market increase with continuous exposure to needs of digitizing in the online world, which has been recently struck to increase exponentially with the pandemic in the past years.

Despite the growth, the video conferencing market has experienced a fair share of challenges. These include meetings being hijacked by online trolls, divulging sensitive information to unauthorized users and the improper management of meeting recordings. Furthermore, a sudden increase in usage is making these platforms vulnerable to various outages like server crashes, data security attacks and platform lags. In March 2020, Microsoft Teams crashed across Europe for two hours, which hit the reputation of Microsoft in the early days of remote working culture. Similarly, the surging usage of Zoom exposed it to severe privacy problems, leading to data security issues and cyber-attacks. As Zoom had been utilizing Chinese servers and was liable to Chinese laws, they were required to disclose all data that resided on their servers. This eventually led to many users migrating to other platforms.

The higher education workforce is no exception to this new work-from-home culture. Not only have business operations become remote, but residential courses have moved online for the foreseeable future. Video conferencing solutions have become an important component of online course delivery.
But these innovative technology solutions introduce new security concerns for higher ed institutions. Are they as safe as they seem?

No, even if we focus on a low scale there are many issues faced in the online education sector. Covid Pandemic /Lockdown has resulted in frustrations among students,It has affected their mental health as well, resulting in unwanted behaviors and activities which disturbs the whole class, and the decorum of the session/academic activities. The links for academic sessions/webinars/CREs are shared by a few students to miscreants, who then login into the meeting/sessions using the same link by using IDs or names of other identified participants. After

login into the meeting, miscreant/mischievous students create indiscipline, confusion and use foul abusive languages to disturb the whole meeting.

While problems such as erroneous student behavior are intrinsic, other problems which exist in the bigger picture when we talk about high level meetings through virtual environments also pose major threats to security, integrity and data vulnerabilities.

Solution

We propose to develop a high-level third party application that can initiate secure connection among various nodes in an organization while solving the problem of on-demand identity management along with removing the issues related to unique tracing down of anomalies and preventing them before they even occur with extra caution tools to be used by management/ organization's tech team throughout the session.

This application will be recognising users with a unique identity for their hardware login that is recognised in the organization's database with the different candidate keys, consisting of different fields like name, phone no., user/registration id e.t.c. This can be used to identify a distinctive login with a registered member while opening the application. This will prevent unauthorized members from using IDs and names of identified participants/students.

After this the IP tracing will be done in the backend autonomously. In order to analyze the incoming connections, we have built a detection methodology that functions as a proxy detection method that intelligently determines how likely an IP address is a proxy/ VPN/ bad IP using advanced mathematical and modern computing techniques. This will return a binary output to the server backend of whether a proxy is being used by the student. If yes, then a prompt will be given to close the VPN in order to access the authentication further. This will greatly reduce any cyber crime/ anomaly in the meetings since without a VPN any crime is fully traceable and risky to commit.

Now that we already know only specific hardware with registered MAC addresses and stable IP addresses can login into the organization's meetings, we have solved the problem of identifying the intruders and miscreants who are trying to join the meet. We will establish methods to improve quality and professionalism inside a meeting.

To carry this out, we have created a machine learning model to filter any NSFW texts that someone types and tries to send. Our application will filter it out beforehand and if it fails the test the backend will redirect the text to the admin and it won't appear in the public chat box after which the admin/host(in this case the professor) will have the power to decide whether to ban the student or not (explained in detail below). This will prevent messages that are usually disturbing and offensive from showing in the chat box and displaying during the session.

Finally, we have built easy to use functions to directly ban the student's registered hardware and logins from the database on a click that a professor can use by sending a post request to the backend to remove the particular id from the database with the time and date and reason to ban. This will force the student to request the administration later with proper apology to be allowed again to attend the classes from next time.

All this will provide utter security from internet breach to classroom discipline throughout an online meeting session.

Methodology

To keep our project as pragmatic as possible, we will be assuming the implementation of a video conferencing application for an organization with dummy nodes in the database. The architecture will be similar to that of any traditional video conferencing app.

The prime ideology of this proposal is hoping to introduce the internet users not only specific to educational institutions but over the world to the cyber-frauds and making them aware of the solution we propose and bring it out to the audience in a large scale through this hackathon.

Technical Implementation/ System Design

Due to the vast scope of our project, we have divided our project into the following parts, which will be explained subsequently with a dummy demonstration -

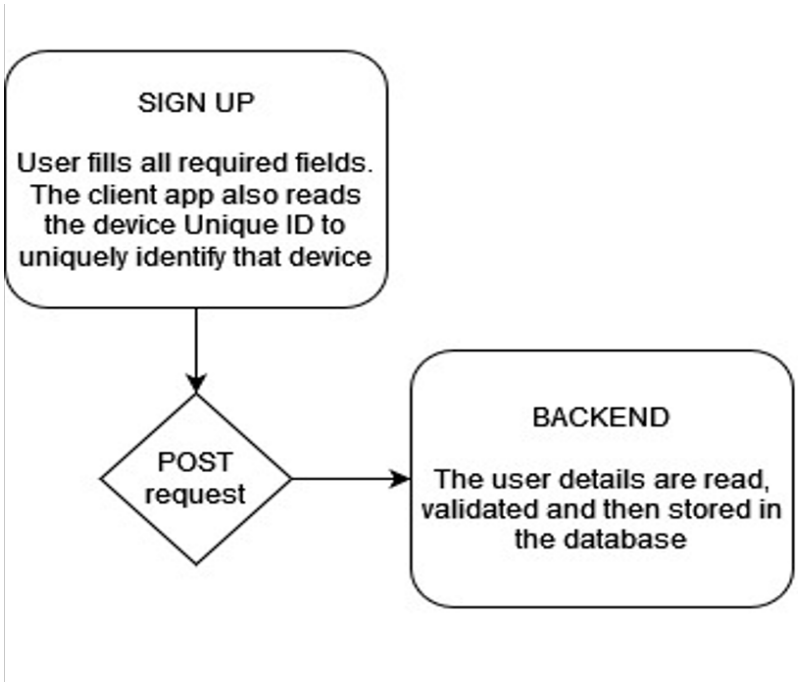
- Setting up our applications and establishing connections and permissions among the various nodes. [organization accessed web app and user centric desktop app with the database and servers to various algorithms in backend.]
- Integrating our application with multiple api calls to the hardwares and running machine learning algorithms to check for the integrity in the connection
- Integrating and facilitating the log check and database handling for the organization to edit and check the backend processes/ servers running behind the meet app.
- Automated NSFW filtering in the video meet conferencing desktop app.
- Provisions to ban and remove hardware access of a member in case of an anomaly from the database with a simple click and easy to understand UI options.

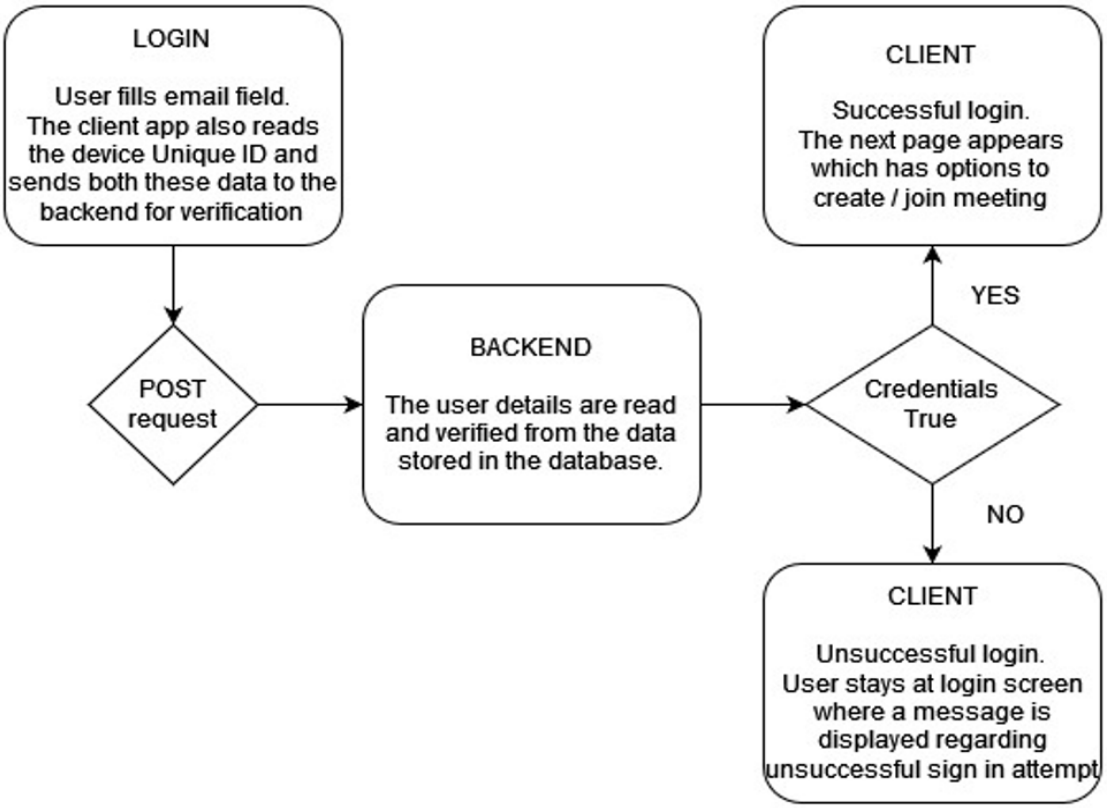
For our demonstration, we will be establishing connections between the various servers to the desktop application (the frontend) with the machine learning, cryptographic and cyber- security algorithms in the backend using a dummy database for a few students and admin logins. Moreover to show the server logs, algorithms and the database status we will be creating a web app that can be accessed by organization-allowed particular logins.

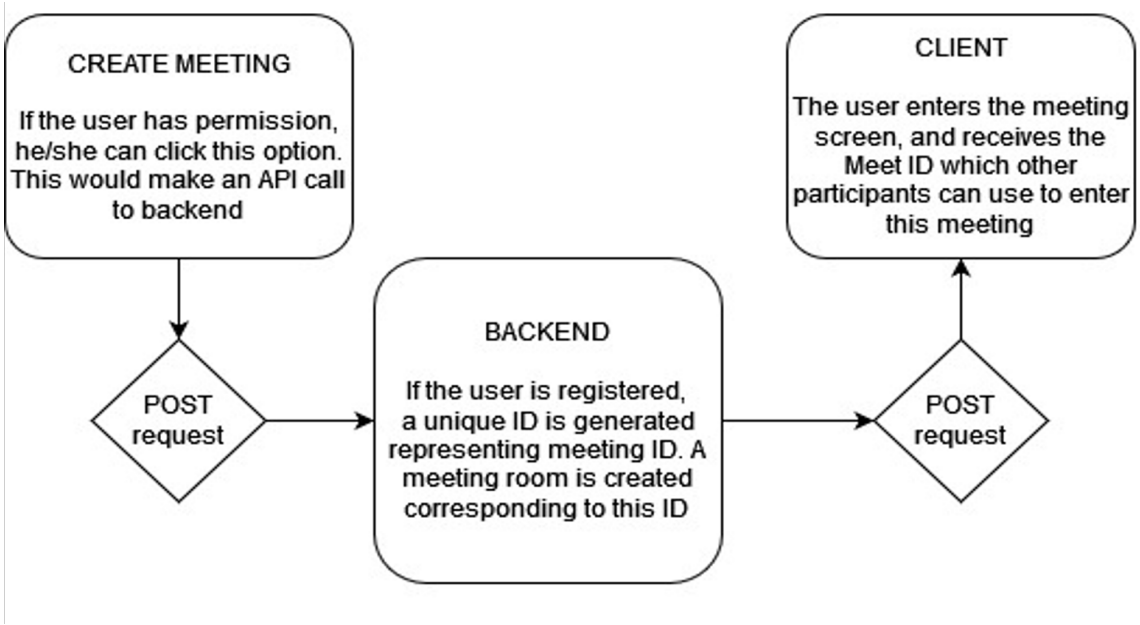
Establishing the Server and Database Backend

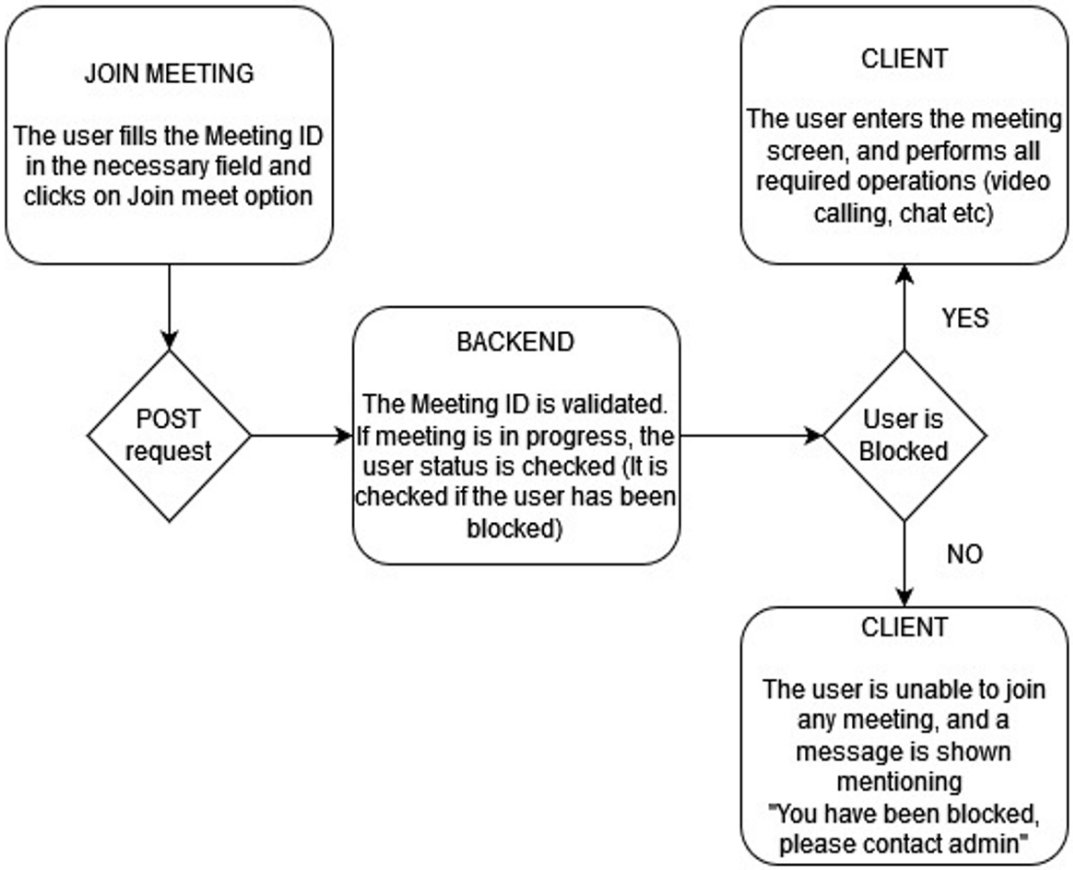
We have used common tech stacks like NodeJS, NPM and express to write the backend of the application with integrated machine learning scripts written in python. This backend is connected to the server and the database which is implemented with MongoDB accessing multiple API calls to and from the client side application.

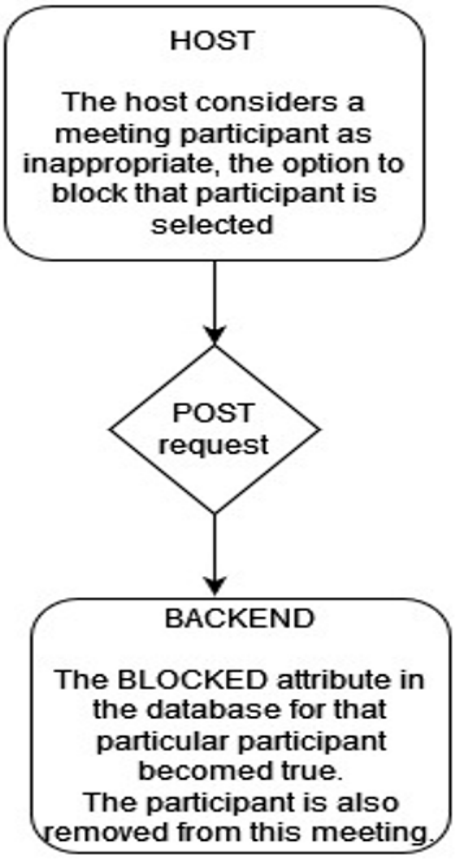
A further architectural overview of these connections is illustrated in the flowchart below.


SIGN UP
User fills all required fields. The client app also reads the device Unique ID to uniquely identify that device
POST request
BACKEND
The user details are read, validated and then stored in the database


LOGIN
User fills email field. The client app also reads the device Unique ID and sends both these data to the backend for verification
POST request
BACKEND
The user details are read and verified from the data stored in the database.
Credentials True
YES
CLIENT
Successful login. The next page appears which has options to create / join meeting
NO
CLIENT
Unsuccessful login. User stays at login screen where a message is displayed regarding unsuccessful sign in attempt


CREATE MEETING
If the user has permission, he/she can click this option. This would make an API call to backend
POST request
BACKEND
If the user is registered, a unique ID is generated representing meeting ID. A meeting room is created corresponding to this ID
POST request
CLIENT
The user enters the meeting screen, and receives the Meet ID which other participants can use to enter this meeting


JOIN MEETING
The user fills the Meeting ID in the necessary field and clicks on Join meet option
POST request
BACKEND
The Meeting ID is validated. If meeting is in progress, the user status is checked (It is checked if the user has been blocked)
User is Blocked
YES
CLIENT
The user enters the meeting screen, and performs all required operations (video calling, chat etc)
NO
CLIENT
The user is unable to join any meeting, and a message is shown mentioning "You have been blocked, please contact admin"



IP Tracing and solving VPN problem

Virtual private networks (VPNs) are becoming a popular method for criminals and other bad actors to hide their online activities. This is helped along by the increase in ease of use of VPNs; they are no longer just a tool for remotely accessing enterprise resources when traveling for work or when working from home. In fact, this could be a use-case for a criminal. If they wish to remotely access an enterprise network in order to steal company and trade secrets, they can use a VPN (or multiple VPNs) in order to hide their own location or to make it appear as if someone else was infiltrating the network.

Proxy connections can be configured in a multitude of fashions. These include configuring a simple redirection within a given browser that will send any web based traffic through the provided proxy service. Alternatively, a proxy connection can be configured as a new network

device, and bridged to the existing network adapter to send packets through a designated server. A client side application can be used, such as TorGuard, to automate the creation and connection type through a designated secure proxy service. Finally, there are websites that act as an anonymous proxy browser by creating a separate frame that connects to the requested sites through a designated server location. Manually configuring a proxy connection requires a fair amount of configuration information including the IP or DNS address, the port being used, the security option utilized for authentication, encryption type, valid user credentials, and the knowledge needed to bridge a network adapter to the configured proxy connection.

Example of a typical proxy

Input:

The proxy check system takes in an input via HTTP GET request. The system fully supports IPv4 with partial support for IPv6.

How it works:

Given an IP address, the system will return a probabilistic value (between a value of 0 and 1) of how likely the IP is a VPN / proxy / hosting / bad IP. A value of 1 means that IP is explicitly banned (a web host, VPN, or TOR node) by our dynamic lists. Otherwise, the output will return a real number value between 0 and 1, of how likely the IP is bad / VPN / proxy, which is inferred through machine learning & probability theory techniques using dynamic checks with large datasets. On average, billions of new records are parsed each month to ensure the datasets have the latest information and old records automatically expire. The system is designed to be efficient, fast, simple, and accurate.

Interpretation of the results:
If a value of 0.50 is returned, then it is as good as flipping a 2 sided fair coin, which implies it's not very accurate. From my personal experience, values > 0.95 should be looked at and values > 0.99 are most likely proxies. Anything below the value of 0.90 is considered as "low risk". Since a real value is returned, different levels of protection can be implemented. It is best for a system admin to test some sample datasets with this system and adjust implementation accordingly. I only recommend automated action on high values (> 0.99 or even > 0.995) but it's good practice to manually review IPs that return high values. For example, mark an order as "under manual review" and don't automatically provision the product for high proxy values. Be sure to experiment with the results of this system before you use it live on your projects. If you believe the result is wrong, don't hesitate to contact me, I can tell you why. If it's an error on my end, I'll correct it. If you email me, expect a reply within 12 hours.

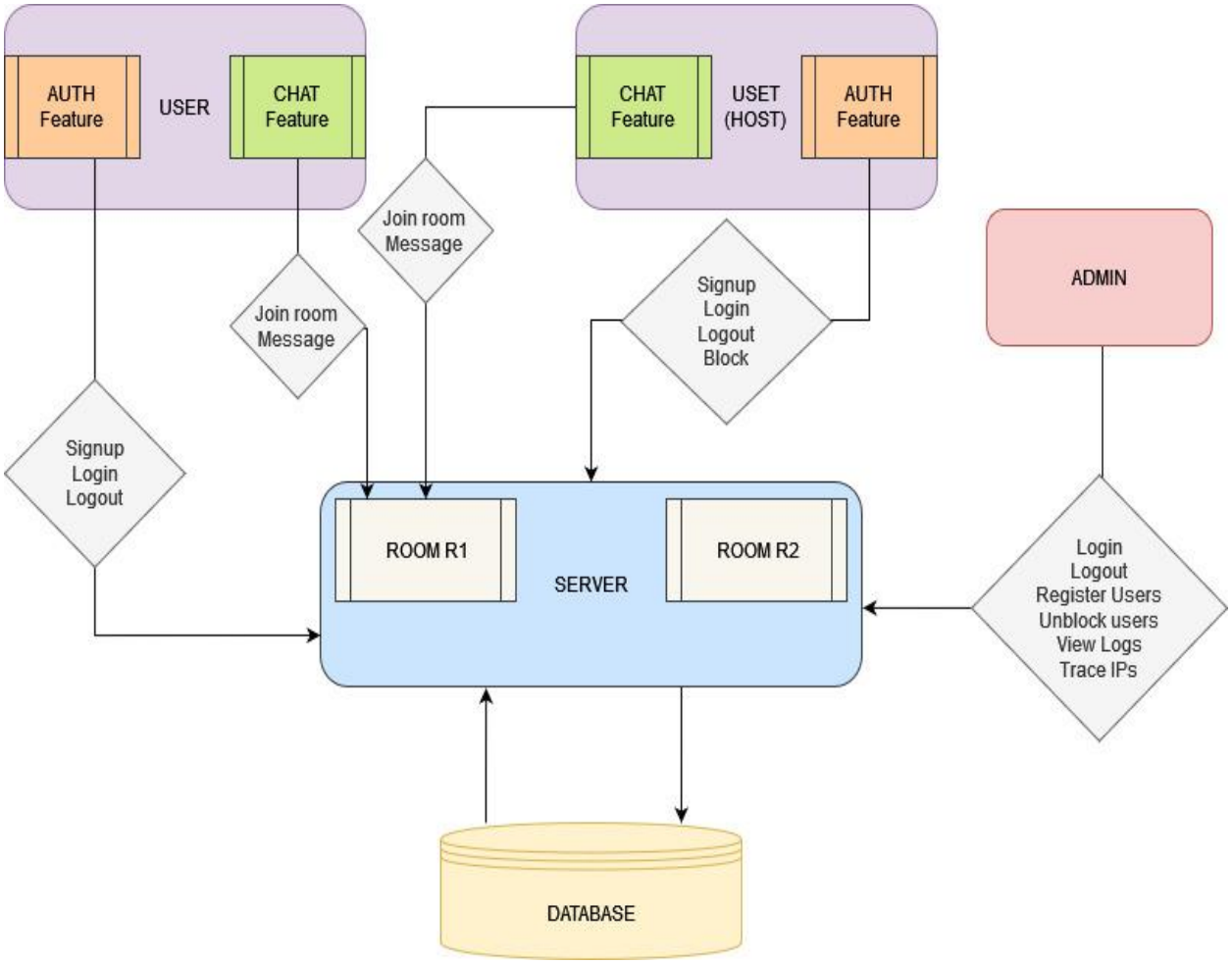
The UI and the filtering algorithms

Keeping in mind the above mentioned complications and methodologies, we are implementing the user interface using electronJS. The application will provide multiple views to user depending on the type of login, whether it is a student login/ professor login or a host login in case of which the application will redirect you to the web app providing further edits and advanced view of the organization's status. For further understanding of the UI please refer to the video demonstration of the application attached with the source code.

Furthermore, we are implementing a text moderation system using NSFW detection deep learning model . We send a post request from our backend server to the hosted model and receive text toxicity status as a response. On the basis of that response, we make the text either available to all (if the status is SFW) and in the other case we make the text only visible to the host or admin of the group marking the text as NSFW, thus providing the admin options for text moderations in chat rooms.

```
{
  "text": "you suck",
  "nsfw_status": [
    {
      "label": "identity_attack",
      "results": [
        {
          "probabilities": {
            "0": 0.9659663438796997,
            "1": 0.034033659845590 59
          },
          "match": false
        }
      ]
    },
    {
      "label": "insult",
      "results": [
        {
          "probabilities": {
            "0": 0.08124697208404541,
            "1": 0.9187529683113098
          },
          "match": true
        }
      ]
    },
    {
      "label": "obscene",
      "results": [
        {
          "probabilities": {
            "0": 0.39931541681289673,
            "1": 0.6006845831871033
          },
          "match": null
        }
      ]
    },
    {
      "label": "severe_toxicity",
      "results": [
        {
          "probabilities": {
```

Integrating the pipeline



Model Significance

Our project is the first of its kind with immense potential to revolutionize the modern banking experience. The promise of flexibility and convenience coupled with transparency and security

make our application quite conspicuous to existing and new customers alike. The user-centric methodology adopted in our application makes it very easy to set up and use.

The problem of unique tracing down of members in a webinar or a meet with restricted logins that we chose to solve is very prominent ignored mainly due to technological constraints. First and foremost, in the present virtual meet scenarios, there is no option for 100% full proof auth check before login, the process of filtration of unnecessary/ unethical texts and the process of tracking down the source of anomalies through multiple participants is very time and labor-intensive as shown. The countless procedures and extra disruption in the virtual calling system make it appalling to have a constant state of mind for other users to carry on with the meeting. Using our application reduces all the unprofessional happenings to 0%.

Furthermore, the use of our application provides promising results in the following factors:

Novelty of the idea:

In terms of novelty, our idea is first of its kind in the video conferencing market. Providing this much security has never been implemented before in any meeting application along with several algorithms to detect connection anomaly and live filtering of unwanted communication.

Complexity :

Our idea ranges from high level complexities from development to implementation of algorithms including integration of machine learning algorithms, creating cryptographic measures to secure databases as well as using cyber security and computer networking concepts to provide integrity to the idea. Moreover multi tech stacks and languages are used to integrate the whole pipeline together to create a final full stack application for direct use.

Clarity and details:

The application is basically a simple desktop + web app with easy to use interface features by clicking simple buttons. Everything else is explained with proper documentation and tutorial inside the github readme files for this project for easy understanding of details. Moreover we are also providing a video tutorial for clarity.

Feasibility:

Being a fully implemented model, we have already overcome all the difficulties which may arise to question the ease of use of our model. The user will just have to download the application and then have a ready to use UI platform with full convenience. Moreover, even the backend part that the organization will have to handle is also implemented in a web app to make the model more feasible for the administration to use in order to handle the database.

Practicability:

One of the most prominent features of our app is practicability as the basic frontend is nothing much different from the traditional conferencing applications. The backend doesn't take much load. Also since the hosting organization carries out the decision making power in case of any unwanted/ mischief act by a student, it is fully practical to add additional security features used by our application, which only makes it stricter for members and more professional for organizations.

Sustainability:

Since the basic functionalities require IP tracing, MAC addresses, official machine learning datasets and simple tech stacks like MongoDB and electronJS, our model is fully sustainable for as long as no new hardware technologies or new cyber attack methods are invented.

Scale of impact:

The scale of impact is in millions expected to cover a market value of 75 billion dollars by 2027, becoming one of the biggest digital industries. Also with the introduction of MR and XR realities, the market for video conferencing will only boost even more and so will the need for security.

User experience:

Our application is convenient, clear and logical to use. It provides full functionality and solves current market problems. Also the implemented web application alongside the main desktop app makes it more promising that the user(hosts) can view all the backend logs and advanced algorithm statistics, further increasing value to user experience.

Societal Impact and future Scope

This model will lead to successful hosting of online meeting sessions and will help society put their trust in the integrity, data safety and decorum of the meetings with the increased factors on security protocols.

Future Scope:

For now we have only implemented a dummy model for a single organization (the database of an educational institute) with dummy data of few logins. For future scope we include the mass usage of this application world wide with millions of users and organizations, proving to be one of the most secure video conferencing applications.Also high level third party APIs can be implemented to be ready to use by many existing applications for the backend features provided with our application.
Few improvements that can be done or features that can be added include intelligent chatbots, improved user interface features, automated note taking, face detection etc.