CS 421.621 Advanced Web APP Development Course

Project Report

Team Name:

# LiveStream

**Team members:**

Karishma Chawla

Kevin Mooney

John Bedingfield

We declare that we have completed this assignment completely and entirely on our own, without any consultation with others. We understand that any breach of the UAB Academic Honor Code may result in severe penalties.
We also declare that the following percentage distribution ***faithfully*** represents individual group members' contributions to the completion of the assignment:

| Name | Contribution (%) | Signature | Date |
|---|---|---|---|
| Karishma Chawla | 33.3 % | | |
| Kevin Mooney | 33.3 % | | |
| John Bedingfield | 33.3 % | jdb | 10 Aug 2020 |

Note: We also utilized the resources listed in the references section, and consulted with Trupeshkumar Patel during his office hours.

# Project LiveStream

## Summary of the Project

Currently, if a user is streaming on multiple platforms (i.e. YouTube Live, Facebook Live, Instagram Live), they have to log-in and manage each stream individually. For example, the user would need to chat on each platform individually.

Project LiveStream integrates chat activities among various streaming platforms. The application allows the user to view an integrated view of all chats, regardless of which platform the chat originated.

Project LiveStream also provides user log-in, an administration capability that allows the user to enter their stream information, and a capability to provide pricing and account information.

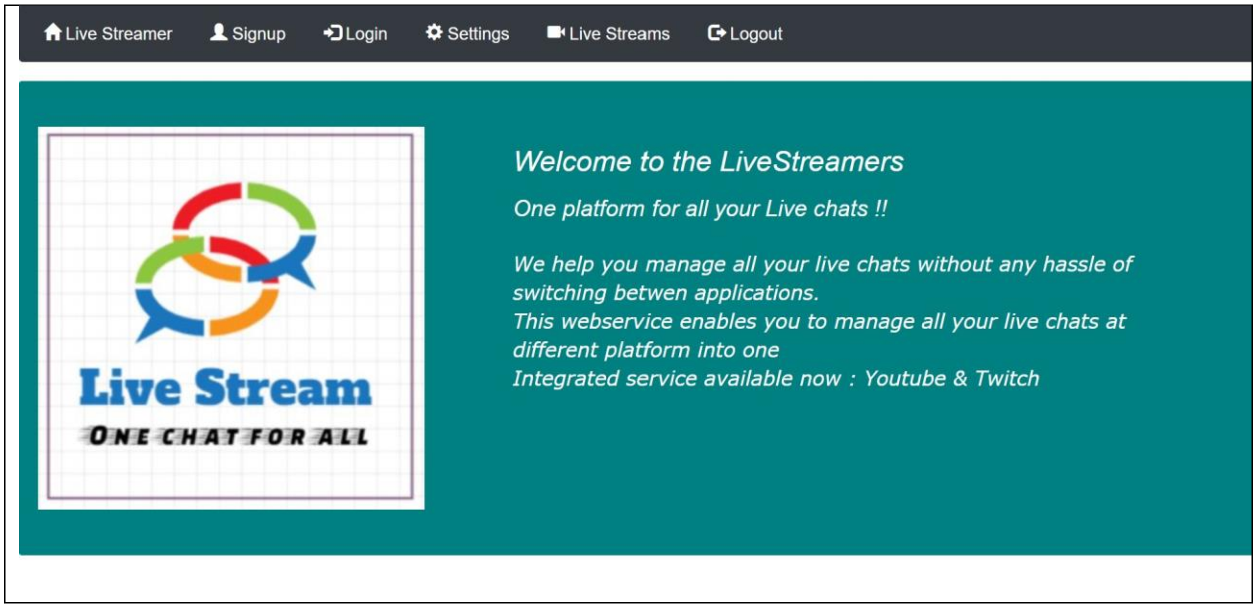
## Technology

For the front-end, HTML, CSS, and Javascript will be the primary technologies used, as well as some bootstrap elements to simplify common front-end look and feel, for example, for the log-in page.
For the back-end, Python and Flask were used.  The database is SQLite, using the Python flask_sqlalchemy library.  In addition, the Flask "session" capability was used for session management, and the following flask capabilities were used throughout the application: request, Response, redirect, url_for.  Finally the APIs for the different streaming platforms were used heavily with various levels of success.

## Project Set up Instructions (readme file)

It is recommended the user first have live streams set-up in YouTube and Twitch.  Therefore, YouTube and Twitch accounts are prerequisites.

1. Set-up a live stream on YouTube and Twitch
2. Since this is a Flask application, start the application with the command "python application.py"

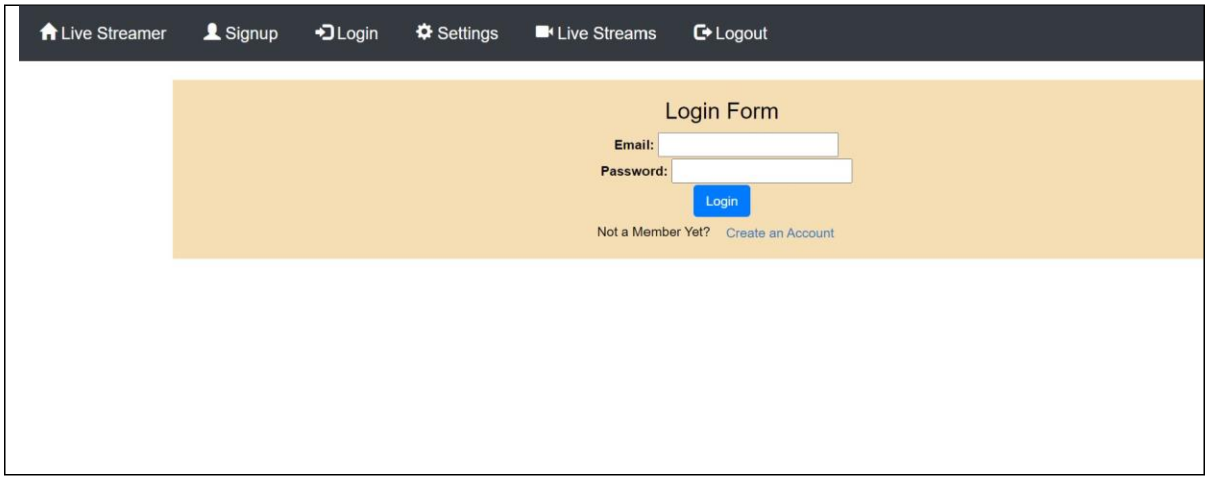


Welcome to the LiveStreamers

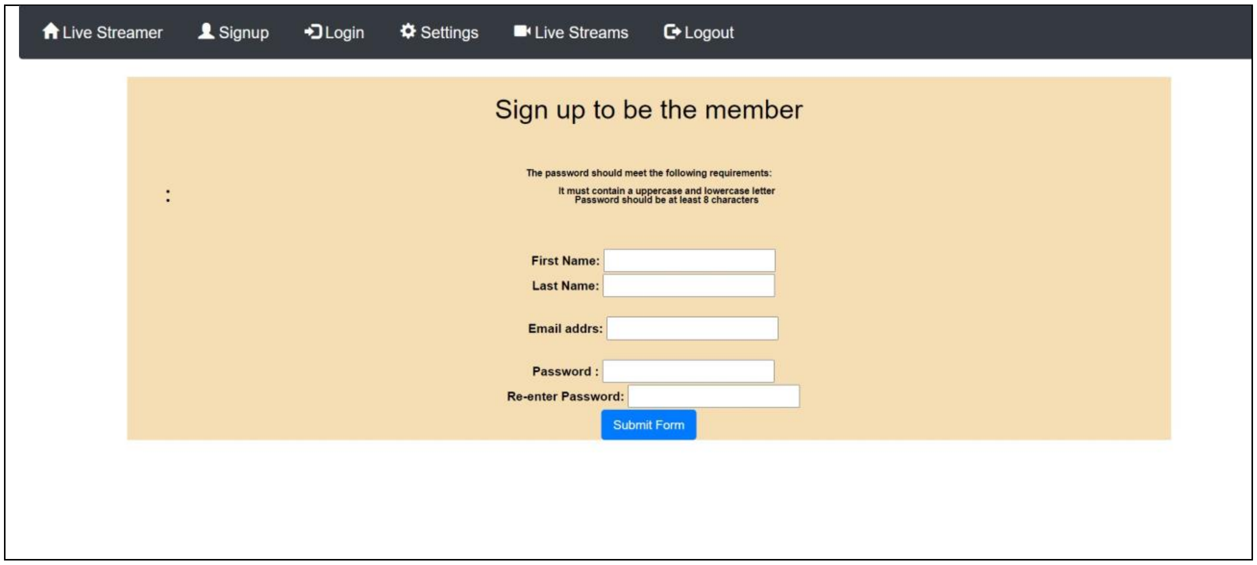
One platform for all your Live chats !!

We help you manage all your live chats without any hassle of switching betwen applications.
This webservice enables you to manage all your live chats at different platform into one
Integrated service available now : Youtube & Twitch

3. Log-in to the LiveStream application



4. If you don’t have a LiveStream account:
    a. Click “sign-up for account”
    b. Enter your user information


Live Streamer Signup Login Settings Live Streams Logout

Sign up to be the member

The password should meet the following requirements:

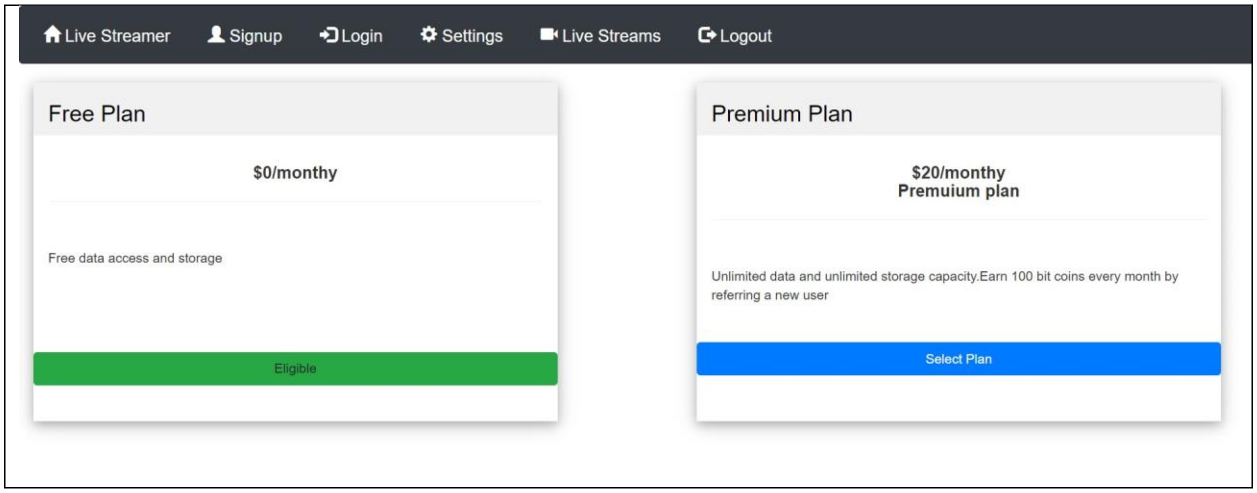
It must contain a uppercase and lowercase letter
Password should be at least 8 characters

:

| | |
|---|---|
| First Name: | |
| Last Name: | |
| Email addrs: | |
| Password : | |
| Re-enter Password: | |

Submit Form

c. Select a Plan


Live Streamer Signup Login Settings Live Streams Logout

Free Plan

$0/monthy

Free data access and storage

Eligible

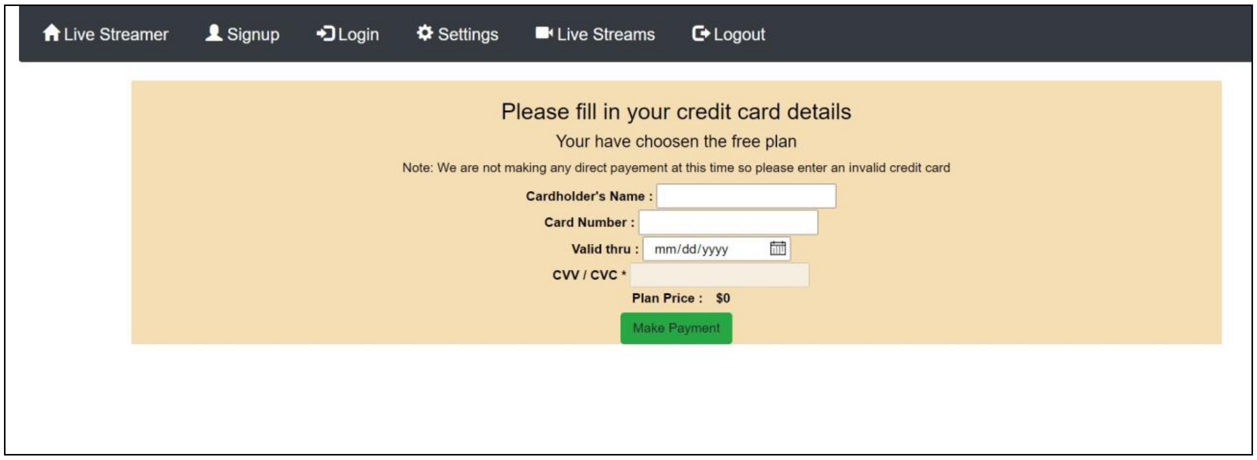
Premium Plan

$20/monthy
Premuium plan

Unlimited data and unlimited storage capacity.Earn 100 bit coins every month by referring a new user

Select Plan

d. Enter billing information



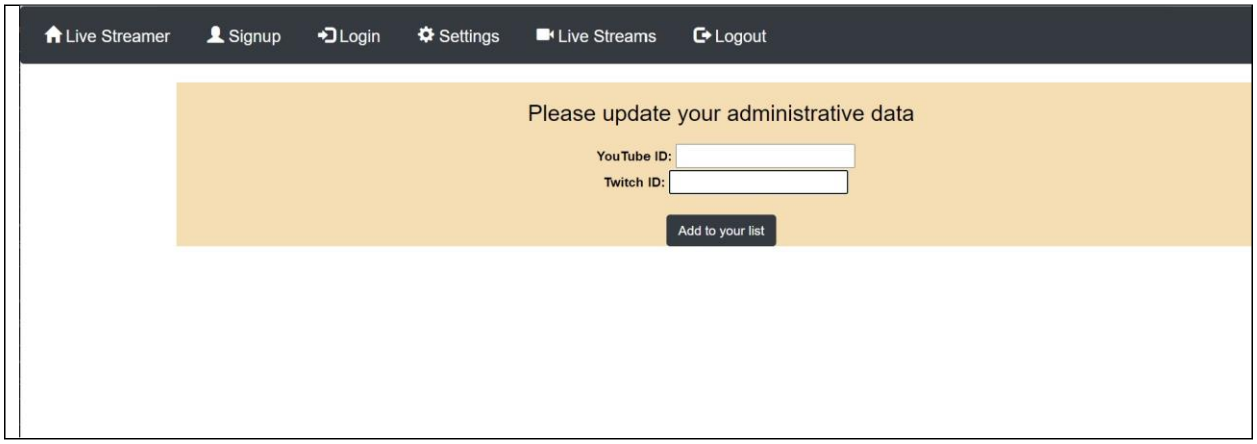
Please fill in your credit card details

Your have choosen the free plan

Note: We are not making any direct payement at this time so please enter an invalid credit card

| | |
|---|---|
| Cardholder's Name : | |
| Card Number : | |
| Valid thru : | mm/dd/yyyy |
| CVV / CVC * | |
| Plan Price : | $0 |

5. Once you are logged into LiveStream, whether you created a new account, or used an existing account, go to the Setup page and enter the YouTube and Twitch stream information



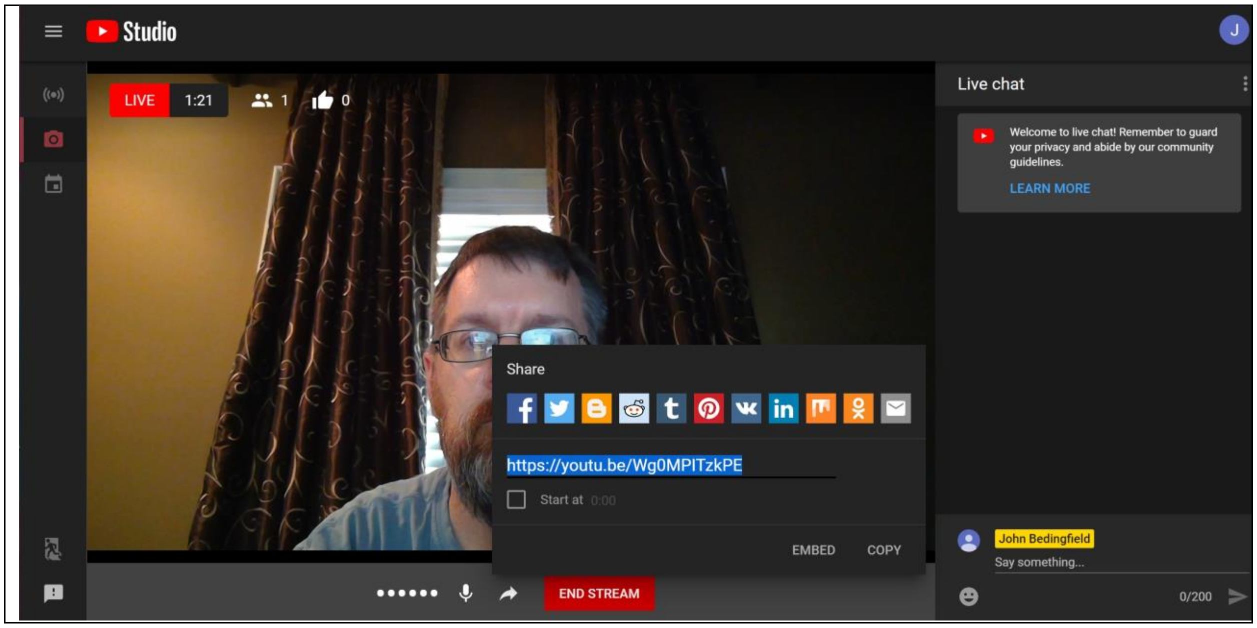
Please update your administrative data

| | |
|---|---|
| YouTube ID: | |
| Twitch ID: | |

a. To find the information in YouTube
   i. In YouTube Studio, next to the live chat there is a “hamburger” click on that and select “pop-out chat”. In the new window’s

url, copy the code after “v=”, it will look something like “Wg0MPITzkPE”

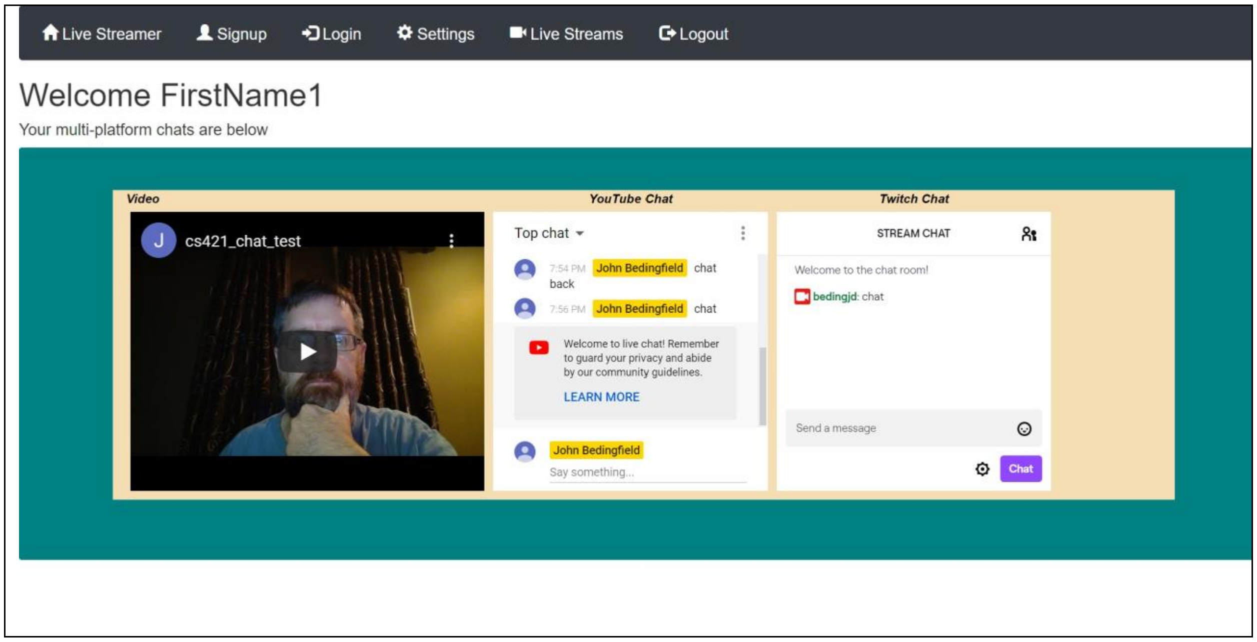
ii. Alternatively, in YouTube Studio, at the bottom of your livestream, there is a white arrow, click that which will lead to a pop-up, the YouTube ID is everything after https://youtu.be/, as shown below, the YouTube ID is “Wg0MPITzkPE”


Studio
LIVE
1:21
1
0
Live chat
Welcome to live chat! Remember to guard your privacy and abide by our community guidelines.
LEARN MORE
Share
https://youtu.be/Wg0MPITzkPE
Start at 0:00
EMBED
COPY
END STREAM
John Bedingfield
Say something...
0/200

b. For Twitch, your ID is your username

6. Go to Stream page to see your streams
   a. On this page you can see the live YouTube video. You can also see the chats from YouTube and Twitch. If you are the one streaming, and you are logging on to the YouTube and Twitch

platforms, you can also chat back (and it will display on those platforms)



Welcome FirstName1

Your multi-platform chats are below

7:54 PM John Bedingfield chat back

7:56 PM John Bedingfield chat

Welcome to live chat! Remember to guard your privacy and abide by our community guidelines.

Welcome to the chat room!

bedingjd: chat

## Challenges

a. Application Programming Interfaces (APIs). The APIs APIs for Facebook, YouTube, and Twitter were sometimes confusing, lacking information, and in some cases did not provide the desired functionality. Perhaps this is because the platforms are incentivized to keep users on their platforms, and not share significant capabilities outside their platforms. Consequently, both chat applications had problems that required solutions not documented in the API.

b. Hosting:  Once the application was working on the team laptops, the code was then pushed to Amazon Web Services (AWS) using the Elastic Beanstalk deployment service.  On AWS neither chat initially worked.  Consequently, trouble shooting and testing were required on the AWS.  Once the problem and solution was determined for the YouTube chat, the solution required a code change.  Subsequently, the new code was specific to the hosting environment, and would not work on the team laptops.  Therefore, the project ended up with one version of the software that works on a "localhost" and one version that works in AWS.  Furthermore, the team was unable to make Twitter chat work in the AWS environment.
c. Newness.  Since the team had no experience in any of the technologies, APIs, web application development, or hosting before the semester, everything was new and many lessons were learned by making and correcting numerous mistakes. Therefore, a lot of time and effect were expended to climb the "learning curve", but the result was that the team learned a lot of useful information that will be practical to future efforts.

# References

## Facebook API references:

https://developers.facebook.com/docs/graph-api/server-sent-events/endpoints/live-comments
https://www.facebook.com/facebookmedia/blog/introducing-the-facebook-live-api

## YouTube API references:

https://developers.google.com/youtube/v3/live/docs/liveChatMessages

Twitch API reference: https://dev.twitch.tv/docs/embed/chat

Flask reference: https://www.flaskapi.org/